PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 5 : A61N 1/08, G08C 15/06

A1

(11) International Publication Number: WO 91/10471

(43) International Publication Date: 25 July 1991 (25.07.91)

(21) International Application Number: PCT/US91/00309

(22) International Filing Date: 15 January 1991 (15.01.91)

(30) Priority data:
468,407 22 January 1990 (22.01.90) US

(71) Applicant: MEDTRONIC, INC. [US/US]; 7000 Central Avenue N.E., Minneapolis, MN 55432 (US).

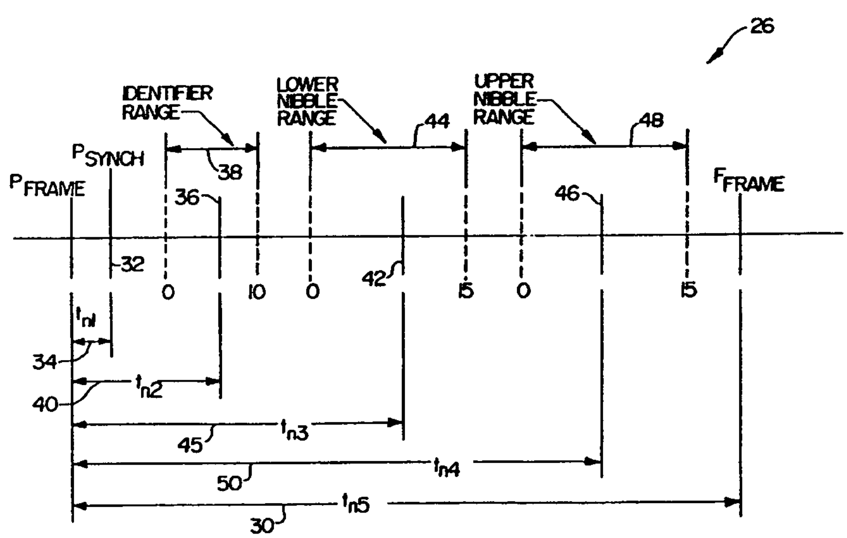
(72) Inventors: WYBORNY, Paul, B. ; 1390 69th Avenue, N.E., Fridley, MN 55432 (US). ROLINE, Glenn, M. ; 4118 211th Lane, N.W., Anoka, MN 55303 (US). NICHOLS, Lucy, M. ; 6781 Marilyn Drive, Maple Grove, MN 55369 (US). THOMPSON, David, L. ; 1660 Onondaga Street, Fridley, MN 55432 (US).

(74) Agents: RISSMAN, John, A. et al.; Medtronic, Inc., 7000 Central Avenue N.E., Minneapolis, MN 55432 (US).

(81) Designated States: AT (European patent), AU, BE (European patent), CA, CH (European patent), DE (European patent), DK (European patent), ES (European patent), FR (European patent), GB (European patent), GR (European patent), IT (European patent), JP, LU (European patent), NL (European patent), SE (European patent).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: IMPROVED TELEMETRY FORMAT



(57) Abstract

A method and apparatus are disclosed for telemetering both analog and digital data from an implantable medical device to an external receiver, such as between an implanted cardiac pacer and its external programming equipment. Analog data is first converted to digital format by an analog-to-digital converter, such that the transmission is digital data. A damped carrier at 175 kilohertz is pulse position modulated by the data. The modulation scheme defines a frame of slightly less than 2 milliseconds. The frame is divided into 64 individual time periods using a crystal clock. The data, along with synchronization and identification codes, are positioned into predefined ranges within each frame as measured by the individual time periods. The data is uniquely identified by the position of a burst of the carrier within the predefined range. This modulation scheme enables necessary data to be transmitted at sufficiently high rates with reduced power requirements thereby conserving the internal battery of the implantable device. This modulation scheme provides flexibility of use, for example, with complex medical devices where transmission of increased volumes of data is desirable, such as cardiac devices having dual-chamber or multisensor capabilities, and for controlling particular conditions, such as tachyarrhythmia.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria
AU	Australia
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CS	Czechoslovakia
DE	Germany
DK	Denmark
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GN	Guinea
GR	Greece
HU	Hungary
IT	Italy
JP	Japan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
LI	Liechtenstein
LK	Sri Lanka
LU	Luxembourg
MC	Monaco
MG	Madagascar
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
NL	Netherlands
NO	Norway
PL	Poland
RO	Romania
SD	Sudan
SE	Sweden
SN	Senegal
SU	Soviet Union
TD	Chad
TG	Togo
US	United States of America

IMPROVED TELEMETRY FORMAT

BACKGROUND OF THE INVENTION

Field of the Invention.

The present invention generally relates to implantable medical devices, and more particularly, pertains to telemetry schemes for percutaneously transmitting analog and digital data from an implantable medical device.

Description of the Prior Art.

The earliest implantable medical devices were designed to operate in a single mode and with no direct percutaneous communication. Later it became clinically desirable to vary certain of the operating parameters and change modes of operation. This was accomplished through the use of programmers and other external devices which transferred commands percutaneously to the implanted medical device.

The communication between the implant and the external world was at first primarily indirect. The operation of an implantable cardiac pacer could be observed, for example, in the electrocardiogram of the patient. Soon it became known that data could be sent from the implanted cardiac pacer by modulating the stimulation pulses in some manner. This can only provide a low bandpass channel, of course, without interfering with the clinical application of the device. Change of the pacing rate to indicate battery condition was a commonly used application of this technique.

As implantable cardiac pacers became more complex, the desirability to transfer more data at higher speeds resulted in the percutaneous transmission of data using a radio frequency carrier. The data to be transmitted is of two basic types, namely, analog and digital. The analog information can include, for example, battery voltage, intracardiac electrocardiogram, sensor signals, output amplitude, output energy, output current, and lead impedance. The digital information can include, for

example, statistics on performance, markers, current values of programmable parameters, implant data, and patient and unit identifiers.

The earliest RF telemetry systems transmitted analog and digital information in separate formats, resulting in inefficient utilization of the available power/bandwidth. Also, these modulation schemes tended to be less than satisfactory in terms of battery consumption, and do not lend themselves to simultaneous transmission of differing data types.

Many types of RF telemetry systems are known to be used in connection with implantable medical devices, such as cardiac pacemakers. An example of a pulse interval modulation telemetry system used for transmitting analog and digital data, individually and serially, from an implanted pacemaker to a remote programmer is disclosed in U.S. Patent No. 4,556,063 issued to Thompson et al., herein incorporated by reference. An example of a modern pacemaker programmer for use with programmable cardiac pacemakers having RF telemetric capabilities is disclosed in U.S. Patent No. 4,550,370 issued to Baker, herein incorporated by reference. However, the telemetry format which is used under these systems, as well as other prior telemetry systems, have not been entirely adequate for reasons described above and a need for significant improvement has continued. As will become apparent from the following, the present invention satisfies that need.

SUMMARY OF THE INVENTION

The present invention percutaneously transmits all data from the implantable medical device in a digital format. It is pulse position modulated on an RF carrier. To accomplish this, the analog quantities must be converted to digital values either at the time of transmission, such as for real-time intracardiac electrocardiograms, or before storage in the memory of

the device, as in the case of historical values of pacing rate for subsequent transmission for trend analysis.

Whether the data to be sent is initially analog or digital, it is transmitted in the same format, i.e., as digital information. The RF carrier is pulse position modulated to conserve battery energy. In this manner, only a short burst of the carrier, e.g., one cycle, is actually needed to transmit a given unit of data. The time position of that burst relative to a synchronizing standard determines the value of the data unit transmitted.

To accomplish this pulse position modulation, a frame of about 2 milliseconds is defined. Within this frame are positioned a synchronizing burst, a frame identifier burst, and one or more data bursts. The synchronizing burst is positioned at a fixed position in the frame. The frame identifier and data are variables, such that the corresponding bursts occur within a range of time within the frame. The range in which a burst is found defines the nature or type of the variable. The position in the range defines the value of the variable.

Because all data transmission is in a digital format, great flexibility is achieved with regard to additional units of data for future applications. The use of the standardized format and capability of encoding more data into a single pulse also decreases the overall battery current requirements and serves to level the energy demand over time. Transmitting the analog data in digital form provides enhanced noise immunity and accuracy.

The transmission protocol provides data rates which are sufficient to transfer clinically useful EGM information in real time. Because each frame is independent, data quantities of varying precision can be transmitted using the same protocol. This modulation scheme provides flexibility of use, for example, with

complex medical devices where transmission of increased volumes of data is desirable in real time, such as cardiac devices having dual-chamber or multisensor capabilities, and for controlling particular conditions such as tachyarrhythmia.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be better understood, and its attendant advantages will be readily appreciated, by reference to the accompanying drawings when taken in consideration with the following detailed description, wherein:

FIG. 1 is a simplified schematic view of an implantable medical device employing the improved telemetry format of the present invention;

FIG. 2 is a conceptual view of one frame of the improved telemetry format of the present invention;

FIG. 3 is a view of the actual transmission pattern of two frames of the improved telemetry format;

FIG. 4 is a block diagram of a portion of an implantable medical device for implementation of the improved telemetry format;

FIG. 5 is a simplified flowchart showing the basic function of software to perform the telemetry uplink operation of the improved telemetry format;

FIG. 6 is a block diagram of the circuitry of the telemetry uplink hardware for implementing the improved telemetry format;

FIG. 7 is a block diagram of the circuitry of the telemetry timing for implementing the improved telemetry format; and

FIG. 8 is a schematic diagram of the driver circuitry for implementing the improved telemetry format.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

A preferred embodiment of the present invention is disclosed relating to use of the improved telemetry format with an implantable cardiac pacer, which may be programmable. However, those of skill in the art will be readily able to adapt the teachings found herein to other implantable medical devices. It will also be understood by those of skill in the art that the telemetry format taught herein can be used for bi-directional communications between an implanted medical device and an external device.

FIG. 1 is a simplified schematic diagram of the present invention as employed in a cardiac pacing system. An implantable pulse generator **10** is implanted in the patient under the outer skin barrier **28**. Implantable pulse generator **10** is electrically coupled to the heart of the patient using at least one cardiac pacing lead **12** in a manner known in the art. Percutaneous telemetry data is transmitted from implantable pulse generator **10** by an RF uplink **26** utilizing the improved telemetry format to a receiving antenna **22**, which is coupled to a programmer **20** via a cable **24**. Receiving antenna **22** also contains a magnet which activates a reed switch in implantable pulse generator **10** as a safety feature, as taught in U.S. Patent No. 4,006,086 issued to Alferness et al., herein incorporated by reference. The telemetry data is demodulated and presented to the attending medical personnel by programmer **20**.

FIG. 2 is a schematic diagram of the protocol of RF uplink **26** using the improved telemetry format. The uplink uses a damped 175 kilohertz RF carrier which is pulse position modulated, as described in detail below. Shown at **30**, the basic timing unit of the format is a frame, having a duration of t_{rs}. It will be understood by those skilled in the art, however, that the present invention can be practiced using fixed-length frames

having periods of shorter or longer duration. In the preferred embodiment, the main timing source of implantable pulse generator **10** comprises a standard 32.768 kilohertz crystal clock which provides a basic clock cycle of 30.52 microseconds. Thus, a frame comprised of 64 clock cycles and extending over a fixed time interval of 1.953125 milliseconds is a convenient frame period, since such frame period is a binary multiple of the basic clock cycle.

A unique synchronizing signal is positioned within a first fixed range of each frame **30**. This signal comprises a synchronizing RF pulse **32** which is located at a time t_{n1} within frame **30**. To properly function as a synchronizing pulse, it must be located at a fixed point within the first fixed range of frame **30**, as shown at **34**.

A four-bit frame identifier code is positioned within a second fixed range of each frame **30**, such second fixed range comprising an identifier range **38**. Identifier range **38** uses a total of eleven basic clock cycles as shown. This identifier code comprises an identifier RF pulse **36** which is pulse position modulated within the identifier range **38**. The position of identifier pulse **36** within identifier range **38** identifies the nature or type of data found within each frame **30** which is being transmitted, such as peak sense, peak pressure, sense threshold and others, as described in further detail below. Shown at **40**, time interval t_{n2} thus uniquely represents the value of identifier pulse **36**, which value in turn identifies the data type being transmitted within frame **30**.

Each frame **30** transfers one eight-bit byte of data along with the identifier code. This data is divided into two portions comprised of four bits of data each. A first portion of this data, namely the four least significant bits of the data byte, is positioned within a third fixed range of frame **30**, such third fixed range

comprising a lower nibble range **44**. A second portion of this data, namely the four most significant bits of the data byte, is positioned within a fourth fixed range of frame **30**, such fourth fixed range comprising an upper nibble range **48**.

A lower nibble pulse **42** is pulse position modulated within lower nibble range **44**, such that its value is uniquely identified by its location, such as at a time t_{n3} shown at **45**. An upper nibble pulse **46** is also pulse position modulated within upper nibble range **48**, such that its value is uniquely identified by its location, such as at a time t_{n4} shown at **50**. Lower nibble range **44** and upper nibble range **48** each comprise sixteen basic clock cycles, permitting each of the sixteen unique values of the four-bit nibble to be specified. To prevent data overlap, suitable guardbands are positioned between each of the ranges within the frame to uniquely identify the synchronizing pulses, thereby avoiding undefined and erroneous data transmission.

FIG. 3 is a diagram of two frames of RF uplink **26**, wherein a first frame corresponds to **Word 1** shown at **70**, and a second frame corresponds to **Word 2** shown at **72**. A count of clock cycles is indicated along an upper horizontal axis of this diagram for each frame. Each basic clock cycle has a duration of 30.52 microseconds. The first frame at **70** is initiated by an RF pulse **52**. A synchronizing RF pulse **54** is shown uniquely identified as precisely four clock cycles later. Because the guardbands are all greater than four clock cycles, no combination of a frame identifier and data can appear as a synchronizing pulse. Synchronizing pulse **54** is used to provide frame synchronization between the transmitter (i.e., implantable pulse generator **10**) and the receiver (i.e., programmer **20**).

An identifier RF pulse **56** is located within identifier range **38**, which range is defined as nine to

nineteen basic clock cycles from the beginning of frame **70**. In **Word 1**, for example, identifier pulse **56** is located at clock cycle nineteen. This identifies the frame as a particular type of data transfer, namely, "Sense Threshold" as indicated in Table 1 below.

TABLE 1

Position	Identification
9	Memory
10	Idle
11	EGM-1
12	Markers
13	Peak Sense
14	Pressure Waveform
15	Peak dp/dt
16	Peak Pressure
17	Delta Capacitor Voltage
18	Activity Counts
19	Sense Threshold

A lower nibble RF pulse **58** is located within lower nibble range **44**, which range is defined as twenty-four to thirty-nine basic clock cycles from the beginning of frame **70**. In **Word 1**, for example, lower nibble pulse **58** is located at clock cycle thirty-one, specifying a binary value of seven on a scale of zero to fifteen. An upper nibble RF pulse **60** is located at clock cycle fifty-eight within upper nibble range **48**, which range is defined as forty-four to fifty-nine basic clock cycles from the beginning of frame **70**, and is demodulated in similar fashion.

FIG. 4 is a block diagram of that portion of implantable pulse generator **10** which is associated with formatting and transmission of RF uplink **26**. Most of the

unique hardware which embodies the present invention is located on a single substrate, being a custom chip device indicated generally by arrow **105**. The remainder is microprocessor-based logic indicated generally by arrow **100**, comprising microprocessor **102**, random access memory (RAM) **104**, and parallel bus **106**. The function of microprocessor-based logic **100** is described in further detail below.

Chip **105** has an analog-to-digital (A/D) converter **108** which receives a number of analog inputs **110** from a multiplexer (not shown). A/D converter **108** permits data to be transferred via RF uplink **26** to be digitized as necessary, so that all data is transmitted in a standardized digital form.

Circuitry (CRC) for generating and analyzing the cyclic redundancy code used to forward error detect telemetry data transmitted over RF uplink **26** is indicated at **112**. In the preferred embodiment, it is also used for data received by implantable pulse generator **10** via a downlink (not shown). Circuitry (DMA) for providing direct memory access to RAM **104** is indicated at **114**, thus permitting multiple byte transfers without constant management by microprocessor **102**.

Key hardware used to implement RF uplink **26** comprises telemetry control and data buffer circuitry indicated generally within dashed lines at **121**, which circuitry includes data buffer **116** and telemetry control **120**, and up-link timing circuitry **118**. Data buffer **116** includes storage for twelve bits of data. This storage is partitioned into a four-bit section **119** for storage of the frame identifier code, and an eight-bit section **117** for storage of the lower nibble and upper nibble of a frame. Data buffer **116** thus stores all of the variables for one complete frame. Data buffer **116** is used to stage the variables for the frame which may be received from

RAM **104**, A/D converter **108**, CRC **112**, or elsewhere along parallel bus **106**.

Telemetry control **120** consists primarily of a telemetry status register. This register stores the telemetry commands and status as loaded by microprocessor **102**. The contents of the register are thus used to gate the data at the proper time of the defined protocol.

Up-link timing **118** decodes the twelve bits of data stored in data buffer **116** to produce a set of timing signals which key bursts of RF energy at the appropriate times to pulse position modulate the 175 kilohertz carrier. Up-link timing **118** also keys bursts of RF energy at the fixed positions within the frame corresponding to the frame-initiating pulse and the synchronizing pulse.

FIG. 5 is a basic flowchart showing the overall function of the microprocessor-based logic **100**. The role is essentially one of initiation of the transfer, rather than management of each detail of the transmission. Software associated with RF uplink **26** is started at element **130**, usually by a down-linked command to transfer data.

Element **132** schedules the requested transmission via the up-link facilities. This scheduling prioritizes uplink transmission requests. Lower priority is given to continuous real time transfers, such as EGM and battery voltage, whereas higher priority is given to single occurrence transmissions of status information.

After scheduling, element **134** determines whether an uplink transmission is currently in progress. If an uplink transmission is in progress, element **132** re-schedules the request.

If an uplink transmission is not in progress after scheduling, element **136** initiates the uplink transmission by activating telemetry control **120**. Exit is via element **138**. While some additional management of the process is

required during the transmission, a description of such further details has been omitted, since it is not believed necessary to one skilled in the art to fully understand the present invention. As to the software associated with the uplink transmission, however, a source code listing of the pertinent sections of such software has been attached hereto as **Appendix A**, and is incorporated by reference herein.

FIG. 6 is a block diagram showing the major data and control signals of telemetry control and data buffer **121** (which includes data buffer **116** and telemetry control **120** shown in **FIG. 4**), and also of up-link timing **118**. A primary function of data buffer **116**, as indicated above, is the staging of the twelve variable bits of a given frame which correspond to a four-bit frame identifier **ID**, and dual-nibble data comprising a four-bit lower nibble **LN** and a four-bit upper nibble **UN**. The data is received over an eight-bit, parallel bus **159** and can be from any one of several sources. Control lines EGMDATA at **150**, PRSDATA at **151**, DLDMA at **153**, DMADS at **155**, LDANDAT at **156**, ENCRC at **161** and LDCRC at **171** specify the source. The output of A/D converter **108** of **FIG. 4** is presented separately to data buffer **116** as an eight-bit parallel transfer to ADC(0-7) at **154** (see **FIG. 6**). The output of CRC **112** is presented separately to data buffer **116** as an eight-bit parallel transfer to CRC(0-7) at **160**, since those devices are located on the same substrate.

Telemetry control **120** outputs a number of control signals, including EGMGAIN at **162**, RVPGAIN at **163**, EGMTELEN at **164**, ANULON at **165**, RAMULON at **166**, MEMEN at **167**, PRSTELEN at **168**, HDRCRCEN at **169** and EGMNPRS at **170**. These control outputs are used to enable and control inputs to data buffer **116**. The key outputs of telemetry control and data buffer **121** are TELRST at **182**, which resets up-link timing **118** and initiates the beginning of a frame, and a parallel data transfer at **184**, which

transfers the frame identifier **ID**, lower nibble **LN** and upper nibble **UN** to up-link timing **118**.

Up-link timing **118** receives the frame-initiating control signal TELRST at **182** and the parallel data transfer (**ID**, **LN** and **UN**) at **184**. A primary function of up-link timing **118** is to key the transmission of 175 kilohertz RF energy at the proper times to indicate start of frame, frame synchronization, frame identifier, lower nibble and upper nibble. Timing for this function is provided by the 32.768 kilohertz crystal clock to up-link timing **118** with clock signal XTAL at **186**. An output TELCLK is provided at **188** which keys the actual burst of RF carrier at the proper times.

FIG. 7 is a block diagram of up-link timing **118**. A frame timing generator **202** provides the desired timing for a frame according to clock input XTAL at **186**, in a manner hereinabove explained. Thus, each frame is comprised of sixty-four basic clock cycles. The process is initiated by receipt of the frame-initiating control signal TELRST at **182**, which enables uplink when in a low state and disables uplink when in a high state. The initial clock cycle of a frame contains a burst of RF energy which is keyed by control signal TELCLK at **188**, which is also used to trigger the start of the data decoding by an uplink word multiplexer **200**.

After the proper four-bit quantity is selected (i.e., frame identifier **ID** first, lower nibble **LN** next, and upper nibble **UN** last), a telemetry pulse timer **204** determines the appropriate timing for a burst to be provided to frame timing generator **202**, and a corresponding burst of RF energy is keyed. Each of the four-bit quantities thus results in the keying of a burst of RF energy at the appropriate time within each frame.

FIG. 8 is a circuit diagram for the drive circuit for generating the RF carrier. A control signal TELCLK at **188** provides the timing information for keying the

carrier. A non-overlap generator **220** functions as a delay device to save current by preventing output transistors **230** and **232** from conducting simultaneously. Every transition of control signal TELCLK at **188** causes one transition by non-overlap generator **220**. Inverters **222**, **224**, **226** and **228** are scaled to provide efficient switching with sufficient drive to the gates of transistors **230** and **232**. Transistors **230** and **232** drive the signal off of chip **105** to ANTDR at **234** to an antenna circuit. A tuned circuit of discreet components, capacitor **236** and coil **238**, are located external to chip **105**. Each transition thus causes this tuned circuit to resonate at 175 kilohertz, thereby generating one uplink burst.

While the invention has been described above in connection with the particular embodiments and examples, one skilled in the art will appreciate that the invention is not necessarily so limited. It will thus be understood that numerous other embodiments, examples, uses and modifications of and departures from the teaching disclosed may be made as to various other systems for telemetering data to and from an implantable medical device, without departing from the scope of the present invention as claimed herein.

APPENDIX A

```
Avocet 6805 Assembler v2.20, #01002   Chip=146805
========== R2 SYSTEM DATA AREA ================= File: DATA.ASM                10/12/89 08:11:23
========== $Revision:   3.0  $ ============                                      Page 9

=0005      ext_tlm_active    EQU    5          ;Extended telemetry is active
=0006      mag_state         EQU    6          ;Magnet state, mode and rate are
                                               ;set to VOO_MODE and mag_rate following
                                               ;permanent programming.
=0007      rr_trans          EQU    7          ;Rate response transition

=0080      TLM_NONMAG_MSK    EQU    10110000B  ;Mask to clear all telemetry
                                               ;flags except those associated
                                               ;with extended telemetry.

           ;******************************************************************
           ;*           tlm2_flags
           ;******************************************************************
=0000      perm_prog_valid EQU      0          ;Valid Permanent programming
                                               ;occurred.
=0001      reset_inhibit    EQU     1          ;Reset inhibit featured
                                               ; - used in validate message
=0002      reset_pace_trigger EQU   2          ;Reset pace trigger featured
                                               ; - used in validate message
=0003      pk_sense_rqst       EQU  3          ;Single Peak sense measurement
                                               ;requested from programmer
=0004      uplnk_cnfrm         EQU  4          ;Uplink confirmation required
                                               ;on next event.


           ;******************************************************************
           ;*.          ULID
           ;******************************************************************
=0005      CRC_error        EQU    5     ;CRC error indicator
=0006      uplink_memory    EQU    6     ;Uplink include memory block
=0007      uplink_CRC       EQU    7     ;Uplink includes CRC and header


           ;******************************************************************
           ;*           Uplink_flags
           ;******************************************************************

=0000      uplnk_disabled EQU      0          ;Uplink is disabled
=0001      uplink_bsy     EQU      1          ;Uplink channel is busy
=0002      up_ram_pnd     EQU      2          ;RAM uplink pending
=0003      up_stat_pnd    EQU      3
=0004      intrrg_pnd     EQU      4          ;Interrogate data uplink pending
=0005      lcap_mrkr_pnd  EQU      5          ;Loss of capture marker uplink
                                              ;pending
=0006      mrkr_pnd       EQU      6          ;Event marker uplink pending
=0007      meas_pnd       EQU      7          ;Measured value uplink pending

=0003      UPLNK_GN_SET   EQU    . (2!^uplnk_disabled + 2!^uplink_bsy)
                                              ;Disable uplink and set busy
                                              ;for gain of signal
           ;******************************************************************
           ;*            Uplink_stat equates
           ;*
           ;******************************************************************

=0004      page0_write        EQU  4          ;Write occured on page 0
=0005      magnet_applied     EQU  5          ;Reed switch is closed
=0006      checksum_error     EQU  6          ;Ram checksum error flag
=0007      POR_occured        EQU  7          ;POR flag

=00F0      UPLNK_CLR_MSK      EQU  11110000B  ;Clear error bits in uplink
                                              ;stat
=00C0      UPLNK_POR_MSK      EQU  11000000B  ;Init mask used during POR

           ;******************************************************************
           ;*
           ;*            Downlink Control Byte equates
           ;*
```

```
Avocet 6805 Assembler v2.20, #01002   Chip=146805                                   10/12/89 08:11:23
========== R2 SYSTEM DATA AREA ================= File: DATA.ASM                              Page 12
========== $Revision:   3.0  $ =============

              ;*****************************************************************
              ;*
              ;*             Telemetry equates
              ;*
              ;*****************************************************************

              ;*****************************************************************
              ;*             Marker values
              ;*****************************************************************

=0066         MK_REFRAC_SENSE EQU     66H              ;Ventrical refractory sense mker
=00EE         MK_SENSE        EQU     0EEH             ;Ventrical sense marker
=00CC         MK_PACE         EQU     0CCH             ;Ventrical pace marker
=0077         MK_LOC          EQU     77H              ;Loss of capture marker
=00DD         MK_TRIGGERED    EQU     0DDH             ;Triggered pace marker

=0080         UP_CRC          EQU     80H              ;Uplink CRC val for ULID regist
=0000         UP_NOCRC        EQU     0                ;Uplink no CRC val for ULID reg
=0040         UP_MEM          EQU     40H              ;Uplink mem val for ULID regist
=0000         UP_NOMEM        EQU     0                ;Uplink no mem val for ULID
                                                       ;register

              ;
              ; ID code and CRC bits for uplink messages
              ;
=0080         STATUS_ID       EQU     0 + UP_CRC + UP_NOMEM    ;Confirmation ID
=00C0         RAM_ID          EQU     0 + UP_CRC + UP_MEM      ;RAM uplink ID
=0043         MARKER_ID       EQU     3 + UP_NOCRC + UP_MEM    ;Marker channel ID
=0044         PKSENSE_ID      EQU     4 + UP_NOCRC + UP_MEM    ;Measure value IDs
=0046         PKDPDT_ID       EQU     6 + UP_NOCRC + UP_MEM
=0047         PKPRESS_ID      EQU     7 + UP_NOCRC + UP_MEM
=0048         DLTAVOLT_ID     EQU     8 + UP_NOCRC + UP_MEM
=0049         ACTCNT_ID       EQU     9 + UP_NOCRC + UP_MEM
=004A         SENSTHRS_ID     EQU     10 + UP_NOCRC + UP_MEM


              ;*****************************************************************
              ;*             Misc. telemetry equates
              ;*****************************************************************

=00C3         ACCESS_CODE     EQU     0C3H             ;Telemetry access code for IPG
=00B3         RM_MODEL_ID     EQU     10110011B        ;IPG model I.D. value, model 8444

=0027         INTRRG_SIZ      EQU     39               ;Size of interrogate block
=0080         MAX_MEMREAD     EQU     128              ;Maximum memory block read size

=000F         PG0             EQU     0FH              ;Control byte Page 0 ID
=0001         PG7             EQU     1                ;Control byte Page 7 ID
=0002         PG8             EQU     2                ;Control byte Page 8 ID
=0004         PG10            EQU     4                ;Control byte Page 10 ID

=0003         DNLK_EXTRA_LEN EQU  3                    ;Message overhead (sub from
                                                       ;HW bytcount)
=0001         DNLK_CB_INDX    EQU  1                   ;First val field in downlink
                                                       ;message
              ;
              ; Emergency values
              ;
=0041         EMG_PW          EQU     41H              ;Emergency Pulse Width (2ms)
=0018         EMG_AMP         EQU     18H              ;Emergency pulse amplitude
                                                       ;(6.0 Volts)

=0023         HIGH_RATE       EQU     23H              ;Highest rate that will allow
                                                       ;full RAM uplink (170ppm)
=001E         UPLINK_DELAY    EQU     1EH              ;Minimum time before next
                                                       ;scheduled event
                                                       ;needed for RAM uplink (300ms)
=0003         UPSTAT_DELAY    EQU     03H              ;Minimum time before next
```

```
Avocet 6805 Assembler v2.20, #01002   Chip=146805                                          10/12/89 08:12:32
========== R2 EXECUTIVE ===================== File: POREXEC.ASM                                      Page 8
========== $Revision:   3.0  $ ==========

;*********************************************************************
;*                        POR and Executive Macros                   *
;*********************************************************************

;@******************** CHECK_MARKER_UPLINK ***********************
;@*                                                                *
;@*  Determine which marker code to uplink while in magnet mode or *
;@*  extended telemetry. If RAM uplink is in progress, the marker  *
;@*  will be igonred.                                              *
;@*                                                                *
;@*  ENTRY CONDITIONS:                                             *
;@*    A pace/sense  or refractory sensed event is being processed. *
;@*    PACESTAT indicates if the event was refractory.             *
;@*                                                                *
;@*  EXIT CONDITIONS:                                              *
;@*    If maker channel is active and a valid marker is detected;  *
;@*    a marker is uplinked.                                       *
;@*                                                                *
;@******************************************************************

;-------------------------------------------------------------------
;@ MACRO CHECK_MARKER_UPLINK
;@    BEGIN
;@      (* check for marker uplink *)
;@      IF (markers_active of mag_flags) THEN
;@        BEGIN
;-------------------------------------------------------------------
;CHECK_MARKER_UPLINK %MACRO
;CMU_START
;                                          ;Jump if marker channel NOT active
;                BRCLR     markers_active,mag_flags,CMU_END
;;------------------------------------------------------------------
;;@          IF ((refractory_evnt of PACESTAT)
;;@            AND (sensed_evnt of exec_flags)) THEN
;;@            BEGIN     (* Refractory sensed event *)
;;@              IF ((timeout_int - event_time) > 1) THEN
;;@                x := MK_REFRAC_SENSE;
;;@              ELSE
;;@                EXIT;
;;@            END;
;;------------------------------------------------------------------
;                                          ;Jump if NOT refractory sensed event
;                BRCLR     refractory_evnt,PACESTAT,CMU_VVT
;                BRCLR     sensed_evnt,exec_flags,CMU_VVT
;                LDA       timeout_int
;                SUB       event_time
;                CMP       #1              ;Is there enough time for marker uplink?
;                BLS       CMU_END         ; No, just exit
;                LDX       #MK_REFRAC_SENSE
;                BRA       CMU_UL          ; Yes, load marker and go uplink it
;;------------------------------------------------------------------
;;@          ELSE IF ((paced_evnt of exec_flags) AND
;;@                  (sensed_evnt of exec_flags)) THEN
;;@            BEGIN
;;@              (* VVT mode, if triggered event send a triggered marker,
;;@                 unless output is inhibited then send sense marker. *)
;;@
;;@              IF NOT(inhibit of tlm_flags) THEN
;;@                x := MK_TRIGGERED;
;;@              ELSE
;;@                x := MK_SENSE;
;;@            END;
;;------------------------------------------------------------------
;CMU_VVT
;                                          ;Jump if NOT both pace and sense
;                BRCLR     paced_evnt,exec_flags,CMU_CKPACE
;                BRCLR     sensed_evnt,exec_flags,CMU_CKPACE
;                                          ;Check for output inhibited
```

```
;               BRCLR   inhibit_enabled,tlm_flags,CMU_INHBT
;               LDX     #MK_SENSE       ;If not, get sense marker
;               BRA     CMU_UL          ;Go uplink it
;CMU_INHBT
;               LDX     #MK_TRIGGERED   ;Else get triggered marker
;               BRA     CMU_UL          ;And send it
;;----------------------------------------------------------------
;;@           ELSE IF ((paced_evnt of exec_flags)
;;@                   AND (NOT(inhibit of tlm_flags))) THEN
;;@             (* If pacing is not inhibited, send a PACE marker.  *)
;;@             x := MK_PACE;
;;----------------------------------------------------------------
;CMU_CKPACE
;                                       ;Jump if NOT paced or if inhibited
;               BRCLR   paced_evnt,exec_flags,CMU_CKSENSE
;               BRSET   inhibit_enabled,tlm_flags,CMU_CKSENSE
;               LDX     #MK_PACE        ;Else get marker code
;               BRA     CMU_UL          ;And send it
;;----------------------------------------------------------------
;;@           ELSE IF (sensed_evnt of exec_flags) THEN
;;@              x := MK_SENSE;
;;@           ELSE
;;@             (* No marker to uplink exit macro *)
;;@             EXIT;
;;----------------------------------------------------------------
;CMU_CKSENSE
;                                       ;Jump if not sensed event
;               BRCLR   sensed_evnt,exec_flags,CMU_END
;               LDX     #MK_SENSE       ;Else get marker value
;;----------------------------------------------------------------
;;@           (* Uplink marker code *)
;;@           CALLM UPLINK_MARKER(x);
;;@         END; (* marker channel active *)
;;@
;;----------------------------------------------------------------
;CMU_UL
;               UPLINK_MARKER           ;Ulink marker (value in x)
;CMU_END
;               %ENDM
;----------------------------------------------------------------
;@    END; (* CHECK_MARKER_UPLINK *)
;----------------------------------------------------------------

$EJECT
```

```
;@********************** UPLINK_MARKER ******************************
;@*                                                                  *
;@*  This macro uplinks event markers if the channel is free.        *
;@*                                                                  *
;@*  ENTRY CONDITIONS:                                               *
;@*    This routine expects x to contain the marker value to be      *
;@*    uplinked.                                                     *
;@*                                                                  *
;@*  EXIT CONDITIONS:                                                *
;@*    If the uplink channel is available it is captured and the     *
;@*    marker is uplinked. If the channel is busy and there are      *
;@*    no pending markers the marker is flagged pending for uplink   *
;@*    at the end of the current uplink.                             *
;@*                                                                  *
;@******************************************************************

;UPLINK_MARKER %MACRO
;
;;-----------------------------------------------------------------
;;@ MACRO UPLINK_MARKER;
;;@   BEGIN
;;@     disable interrupts;
;;@     (* Check if uplink channel is available *)
;;@     IF NOT(uplnk_disabled of uplink_flags) THEN
;;@       BEGIN
;;@         IF NOT(uplink_bsy of uplink_flags) THEN
;;@           BEGIN
;;@             (* If Uplink channel is free then uplink marker *)
;;@             uplink_bsy of uplink_flags := TRUE;
;;@             enable interrupts;
;;@             marker_val := x;
;;@             TELADHI := HIADDR(marker_val);
;;@             TELADLO := LOADDR(marker_val);
;;@             BYTCOUNT := 1;
;;@             ULID := MARKER_ID;
;;@             RAM_uplink of TELSTAT := TRUE;
;;@           END;
;;-----------------------------------------------------------------
;UPM_START
;               SEI                         ;Disable interrupts
;                                           ;Jump if uplink disabled
;               BRSET     uplnk_disabled,uplink_flags,UPLMDONE
;UPLMARKER
;                                           ;Jump if uplink BUSY
;               BRSET     uplink_bsy,uplink_flags,UPL_BSY
;;
;;              Uplink NOT busy
;;
;                                           ;Flag uplink busy
;               BSET      uplink_bsy,uplink_flags
;               CLI                         ;Enable interrupts
;               STX       marker_val        ;Put marker value in buffer
;               LDA       #HIGH marker_val ;Get MSB of buffer address
;               STA       TELADHI           ;Write it to hardware
;               LDA       #LOW marker_val ;Get LSB of buffer address
;               STA       TELADLO           ;Etc.
;
;               LDA       #1                ;Get output count
;               STA       BYTCOUNT          ;Write to hardware count register
;
;               LDA       #MARKER_ID        ;Get ID code
;               STA       ULID              ;Tell the hardware
;                                           ;Start the uplink
;               BSET      RAM_uplink,TELSTAT
;               BRA       UPLMDONE
;;-----------------------------------------------------------------
;;@         ELSE
;;@           BEGIN
```

```
Avocet 6805 Assembler v2.20, #01002   Chip=146805                                   10/12/89 08:12:32
========== R2 EXECUTIVE ===================== File: POREXEC.ASM                            Page 13
========== $Revision:   3.0  $ ==========

;;@              (* If no markers are pending the flag one pending *)
;;@              mrkr_pnd of uplink_flags := TRUE;
;;@              marker_val := x;
;;@            END;
;;@        END;
;;@      enable interrupts;
;;@
;;-------------------------------------------------------------------
;; Uplink BUSY
;;
;UPL_BSY
;               BSET      mrkr_pnd,uplink_flags    ;Flag marker pending and
;               STX       marker_val               ;store marker in the buffer
;
;UPLMDONE
;               CLI                           ;Enable interrupts
;               %ENDM
;-------------------------------------------------------------------
;@  END;    (* UPLINK_MARKER *)
;-------------------------------------------------------------------

$EJECT
```

```
;@********************** UPLINK_INTRRG *******************************
;@*                                                                  *
;@*  This macro uplinks the interrogate block of size INTRRG_SIZ and  *
;@*  starting at the address pointed to by INTRRG_AD if the uplink   *
;@*  channel is free. Otherwise, if there is no RAM uplink, the      *
;@*  interrogate block is set pending and is scheduled via the next  *
;@*  TELBF interrupt, occurring when the uplink channel becomes      *
;@*  free. All other uplinks have to be disabled while checking the  *
;@*  uplink flags to avoid contention of the uplink channel.         *
;@*                                                                  *
;@*  ENTRY CONDITIONS:                                               *
;@*    None.                                                         *
;@*                                                                  *
;@*  EXIT CONDITIONS:                                                *
;@*    None.                                                         *
;@*                                                                  *
;@*******************************************************************

;-------------------------------------------------------------------
;@MACRO UPLINK_INTRRG;
;@BEGIN
;@ (* Capture uplink channel - If busy set interrogate pending *)
;@  disable interrupts;
;@  IF NOT(uplnk_disabled of uplink_flags) THEN
;-------------------------------------------------------------------
;UPLINK_INTRRG %MACRO
;               SEI                        ;Dissable interrupts
;               BRSET     uplnk_disabled,uplink_flags,UI_END
;;------------------------------------------------------------------
;;@    BEGIN
;;@      IF NOT(uplink_bsy of uplink_flags) THEN
;;@        BEGIN
;;@          uplink_bsy of uplink_flags := TRUE;
;;@          enable interrupts;
;;@          statbyt := uplnk_stat;
;;@          CALLN LOAD_INTRRG_UPLINK WITHIN R2LIB;
;;@          RAM_uplink of TELSTAT := TRUE;
;;@        END;
;;------------------------------------------------------------------
;               BRSET     uplink_bsy,uplink_flags,UI_UBSY
;               BSET      uplink_bsy,uplink_flags
;               CLI                        ;Enable interrupts
;               LDA       uplnk_stat
;               STA       statbyt          ;Initialize the uplink status byte
;UI_LIU
;               LOAD_INTRRG_UPLINK
;UI_LIU_END
;               BSET      RAM_uplink,TELSTAT
;               BRA       UI_END
;;------------------------------------------------------------------
;;@      ELSE
;;@        intrrg_pnd of uplink_flags := TRUE;
;;@    END;
;;------------------------------------------------------------------
;UI_UBSY
;               BSET      intrrg_pnd,uplink_flags
;;------------------------------------------------------------------
;;@    enable interrupts;
;;@
;;------------------------------------------------------------------
;UI_END
;               CLI                        ;Enable interrupts
;               %ENDM
;-------------------------------------------------------------------
;@END;   (* UPLINK_INTRRG *)
;-------------------------------------------------------------------

$EJECT
```

```
Avocet 6805 Assembler v2.20, #01002   Chip=146805                                    10/12/89 08:15:30
========== R2 LSCAP INTERRUPT MODULE ============ File: LOC.ASM                            Page 10
========== $Revision:   3.0  $ ============
```

```
;@********************** UPLINK_LCAP_MARKER ***********************
;@*                                                               *
;@*  This macro uplinks loss of capture markers.                  *
;@*                                                               *
;@*  ENTRY CONDITIONS:                                            *
;@*    Under magnet operations, the LSCAPINT interrupt is used for *
;@*    the uplink of LOC markers if the channel is free.          *
;@*                                                               *
;@*  EXIT CONDITIONS:                                             *
;@*    None.                                                      *
;@*                                                               *
;@*****************************************************************

;-----------------------------------------------------------------
;@  MACRO UPLINK_LCAP_MARKER;
;@  BEGIN
;@
;@    disable interrupts;
;@    IF NOT (uplnk_disabled of uplink_flags) THEN
;-----------------------------------------------------------------
;UPLINK_LCAP_MARKER %MACRO
;ULM_START
;               SEI                         ;Disable interrupts
;ULM_INT
;                                           ;Jump if NOT (NOT uplnk_disabled)
;               BRSET    uplnk_disabled,uplink_flags,ULM_DONE
;
;;----------------------------------------------------------------
;;@        BEGIN
;;@          IF NOT(uplink_bsy of uplink_flags) THEN
;;----------------------------------------------------------------
;                                           ;Jump if uplink busy
;               BRSET    uplink_bsy,uplink_flags,ULM_LCP
;
;;----------------------------------------------------------------
;;@            BEGIN
;;@              (* If Uplink channel is free then uplink marker *)
;;@              uplink_bsy of uplink_flags := TRUE;
;;@              enable interrupts;
;;@              TELADHI := HIADDR(LCAP_MARKER);
;;@              TELADLO := LOADDR(LCAP_MARKER);
;;@              BYTCOUNT := 1;
;;@              ULID := MARKER_ID;
;;@              RAM_uplink of TELSTAT := TRUE;
;;@            END;
;;----------------------------------------------------------------
;               BSET     uplink_bsy,uplink_flags
;               CLI                         ;Enable interrupts
;               LDA      #HIGH lcap_marker ;Get address MSB
;               STA      TELADHI            ;Write to controller register
;
;               LDA      #LOW lcap_marker ;Get address LSB
;               STA      TELADLO            ;Write to controller
;               LDA      #1                 ;Get byte count
;               STA      BYTCOUNT           ;Write to controller
;               LDA      #MARKER_ID         ;Get ID
;               STA      ULID               ;Write to controller
;               BSET     RAM_uplink,TELSTAT ;Start uplink
;               BRA      ULM_DONE           ;Thats all folks
;
;;----------------------------------------------------------------
;;@          ELSE
;;@            BEGIN
;;@              (* If no markers are pending the flag one pending *)
;;@              lcap_mrkr_pnd of uplink_flags := TRUE;
;;@            END;
;;@        END;
;;----------------------------------------------------------------
```

```
;ULM_LCP
;                                       ;Jump if lcap marker pending
;               BSET    lcap_mrkr_pnd,uplink_flags
;
;;-----------------------------------------------------------------
;;@     enable interrupts;
;;-----------------------------------------------------------------
;ULM_DONE
;               CLI                     ;Enable interrupts
;
;               %ENDM

;-----------------------------------------------------------------
;@
;@  END;    (* UPLINK_LCAP_MARKER *)
;@
;-----------------------------------------------------------------

$EJECT
```

```
;***********************************************************************
;*                        ADC Interrupt Macros                         *
;***********************************************************************

;@********************** UPLINK_MEAS_VAL ******************************
;@*                                                                    *
;@*  This macro is used to uplink measured values.                     *
;@*                                                                    *
;@*  ENTRY CONDITIONS:                                                 *
;@*    The array meas_val has been loaded with the appropriate         *
;@*    data for uplink. The number of bytes for uplink is stored       *
;@*    in the x register.                                              *
;@*                                                                    *
;@*  EXIT CONDITIONS:                                                  *
;@*    If the uplink channel is free it is captured and the data       *
;@*    in the meas_val buffer is uplinked. If the uplink channel       *
;@*    is busy with a RAM uplink the measured values are               *
;@*    discarded. Otherwise if the channel is busy the measured        *
;@*    values are flagged as pending and uplinked on the next          *
;@*    TELBF interrupt.                                                *
;@*                                                                    *
;@*********************************************************************

;---------------------------------------------------------------------
;@  MACRO UPLINK_MEAS_VAL(x);
;@  BEGIN
;@    IF NOT(uplnk_disabled of uplink_flags) THEN
;---------------------------------------------------------------------
;UPLINK_MEAS_VAL %MACRO
;UMV_START
;             BRSET    uplnk_disabled,uplink_flags,UMV_END
;
;;--------------------------------------------------------------------
;;@       BEGIN
;;@         IF NOT(uplink_bsy of uplink_flags) THEN
;;--------------------------------------------------------------------
;                                    ;Jump if uplink busy
;             BRSET    uplink_bsy,uplink_flags,UMV_SMV
;
;;--------------------------------------------------------------------
;;@           BEGIN
;;@            (* Uplink channel free uplink measured value buffer *)
;;@             uplink_bsy of uplink_flags := TRUE;
;;@             TELADHI := HIADDR(meas_val[0]);
;;@             TELADLO := LOADDR(meas_val[0]);
;;@             BYTCOUNT := x;
;;@             ULID := meas_id;
;;@             RAM_uplink of TELSTAT := TRUE;
;;@           END;
;;--------------------------------------------------------------------
;             BSET     uplink_bsy,uplink_flags ;Set uplink busy
;             LDA      #HIGH meas_val ;Get buffer address MSB
;             STA      TELADHI        ;Write DMA address register
;
;             LDA      #LOW meas_val  ;Get buffer address LSB
;             STA      TELADLO        ;etc.
;             STX      BYTCOUNT       ;Write byte count
;             LDA      meas_id        ;Get ID
;             STA      ULID           ;Write to hardware
;             BSET     RAM_uplink,TELSTAT ;Start uplink
;             BRA      UMV_END        ;Go exit
;
;;--------------------------------------------------------------------
;;@         ELSE  (* NOT uplink_bsy *)
;;@           BEGIN
;;@            (* Set measured value uplink pending *)
;;@             meas_count := x;
;;@             meas_pnd of uplink_flags := TRUE;
;;@           END;
;;@       END;
;;--------------------------------------------------------------------
;
```

```
;UMV_SMV
;               STX     meas_count      ;Save pending byte count
;               BSET    meas_pnd,uplink_flags ;Show pending upling
;               BRA     UMV_END         ;Thats all folks
;UMV_END
;               XENDM
;--------------------------------------------------------------------
;@  END;    (* UPLINK_MEAS_VAL *)
;--------------------------------------------------------------------

$EJECT
```

```
;@****************************** TLM ***********************************
;@*                                                                    *
;@*    R2, Pacemaker Model 8444                                        *
;@*    MODULE:       TLM                                               *
;@*                                                                    *
;@*    The TLM module processes magnet mode operations while the reed *
;@*    switch is closed. These include the handling of the telemetry  *
;@*    protocol, the TMT and lead test activation, the pulse pressure *
;@*    calculation for loss of capture markers detection. The         *
;@*    telemetry protocol involves processing downlink and uplink     *
;@*    messages. Downlink messages are validated before being acted   *
;@*    upon. The uplink consists of confirmation and confirmation +   *
;@*    replies to downlink requests.                                  *
;@*                                                                    *
;@*    Routines defined in this module include:                       *
;@*                                                                    *
;@*       Macros:                                                      *
;@*          DO_MEMWRITE            - transfer downlink record to     *
;@*                                   memory                          *
;@*          EXEC_SPEC_FUNC         - decode and execute special      *
;@*                                   function                        *
;@*          EXEC_SPEC_REQ          - decode and execute special      *
;@*                                   requests                        *
;@*          PROCESS_MEMWRITE       - transfer downlink record to     *
;@*                                   memory     and evaluate it        *
;@*          PROCESS_MSG            - decode memory offsets           *
;@*          SWITCH_TO_NON_MAGMODE - restore non_magnet mode         *
;@*                                   operation                       *
;@*          VALIDATE_MSG           - validate downlink message       *
;@*                                                                    *
;@*       Procedures:                                                  *
;@*          None.                                                     *
;@*                                                                    *
;@*       Drivers:                                                     *
;@*          GNLSINT_PROC           - gain or loss interrupt handler  *
;@*          RDSWINT_PROC           - reed-switch interrupt handler   *
;@*          TELBFINT_PROC          - telemetry buffer interrupt      *
;@*                                   handler                         *
;@*                                                                    *
;@**********************************************************************


            DEFSEG    TLM,CLASS=CODE
            SEG       TLM

$SETLN(MACROS.INC);              %INCLUDE  "MACROS.INC"
```

```
$NOALLPUBLIC
$SETLN(EQUATES.INC);                %INCLUDE  "EQUATES.INC"
```

```
;***********************************************************************
;*                         Telemetry Subroutines                       *
;***********************************************************************

$EJECT
```

```
Avocet 6805 Assembler v2.20, #01002   Chip=146805                                   1/18/90 10:35:56
========== R2 TELEMETRY MODULE ================= File: TLM.ASH                                Page 34
========== $Revision:   3.3  $ =============

                  ;*********************************************************************
                  ;*                    Telemetry Interrupt Handlers                    *
                  ;*********************************************************************
                  ;@************************* GNSLINT_PROC ****************************
                  ;@*                                                                 *
                  ;@*  This procedure is the gain/loss interrupt handler and it is    *
                  ;@*  non-preemptive. It is responsible for controlling the downlink *
                  ;@*  and disabling uplink. Whether the interrupt is do to a gain or a *
                  ;@*  loss of signal can be determined by reading a bit in the TELSTAT *
                  ;@*  register. At the beginning of a downlink all pending uplinks are *
                  ;@*  abandoned and the TELBF interrupt is masked out until the end of *
                  ;@*  downlink. In which case it is reenabled, after being first    *
                  ;@*  cleared, in the case of downlink overrun. Downlink is then     *
                  ;@*  disabled until just before the uplink response, either a status *
                  ;@*  uplink or a RAM uplink.                                        *
                  ;@*                                                                 *
                  ;@*  ENTRY CONDITIONS:                                              *
                  ;@*    No other interrupts are enabled at this point, ADC interrupts *
                  ;@*    are the only higher priority and they are ignored during     *
                  ;@*    telemetry.                                                   *
                  ;@*                                                                 *
                  ;@*  EXIT CONDITIONS:                                               *
                  ;@*    None.                                                        *
                  ;@*                                                                 *
                  ;@*******************************************************************

                  ;-------------------------------------------------------------------
                  ;@PROCEDURE GNLSINT_PROC;
                  ;@BEGIN
                  ;@
                  ;@  (* Check if gain or loss of signal occurred. *)
                  ;@  IF (downlnk_present of TELSTAT) THEN
                  ;@    BEGIN
                  ;@      (* Gain of downlink signal. Clear pending uplinks, disable
                  ;@         uplink and TELBF interrupts, and clear any ADC and
                  ;@         TELBFINT interrupts.                                  *)
                  ;@      uplink_flags := UPLNK_GN_SET;
                  ;@      IF (TMT of mag_flags) THEN
                  ;@        reset_TMT of mag_flags := TRUE;
                  ;@      TELBFINT of ipgstate_msk := TRUE;
                  ;@      IRQREG := TELBFINT_MSK;
                  ;@      ULID := 0;
                  ;@
                  ;@      (* If POS currently executing then postpone loss-of-signal
                  ;@         processing until after POS is complete.               *)
                  ;@      IF ((sensed_evnt of exec_flags)
                  ;@          OR (paced_evnt of exec_flags)) THEN
                  ;@        GNLSINT of current_pri := TRUE;
                  ;@    END;
                  ;@
                  ;-------------------------------------------------------------------
                  GNLSINT_PROC
0000& 09 00* 1D                 BRCLR    downlnk_present,TELSTAT,GNLS_LOSS
0003& A6 03                     LDA      #UPLNK_GN_SET
0005& B7 00*                    STA      uplink_flags        ;Disable uplink
0007& 09 00* 02                 BRCLR    TMT,mag_flags,GNLS_NTMT
000A& 1A 00*                    BSET     reset_TMT,mag_flags ;Reset TMT if active
                  GNLS_NTMT
000C& 14 00*                    BSET     TELBFINT,ipgstate_ms     ;Mask TELBF interrupts
000E& AE 04                     LDX      #TELBFINT_MSK
0010& BF 00*                    STX      IRQREG                   ;Clear TELBF interrupts
0012& 4F                        CLRA
0013& B7 00*                    STA      ULID                     ;Clear ULID register
0015& B6 00*                    LDA      exec_flags               ;Is POS currently executing?
0017& A4 03                     AND      #((1 SHL sensed_evnt) + (1 SHL paced_evnt))
0019& 27 02                     BEQ      GNLS_NPOS                ; No, then exit
001B& 12 00*                    BSET     GNLSINT,current_pri      ; Yes, postpone loss-o-signal
                                                                  ;      until after POS
```

```
                  GNLS_NPOS
001D& CC 0384&                      JMP       GNLS_DONE
                  ;-------------------------------------------------------------------
                  ;@ ELSE IF (uplnk_disabled of uplink_flags) THEN
                  ;@   BEGIN
                  ;@     (* If the uplnk_disabled bit was not set then a downlink
                  ;@        overrun has occured (gain of signal was missed) and
                  ;@        downlink should be ignored!!                          *)
                  ;@
                  ;@     IF (reset_TMT of mag_flags) THEN
                  ;@       CALL TMT_RESET WITHIN R2LIB;
                  ;@
                  ;@     uplnk_stat := (uplnk_stat AND UPLNK_CLR_MSK);
                  ;@
                  ;@     IF (TELBFINT of IRQREG) THEN
                  ;@       BEGIN
                  ;@         (* Downlink overflow - Flag error, uplink status,
                  ;@            and clear TELBF interrupt *)
                  ;@         IRQREG := TELBFINT_MSK;
                  ;@         up_stat_pnd of uplink_flags := TRUE;
                  ;@         uplnk_stat := uplnk_stat OR DNLK_OVF_ERR;
                  ;@       END;
                  ;-------------------------------------------------------------------
                  GNLS_LOSS
0020& 00 00* 03                     BRSET     uplnk_disabled,uplink_flags,GNLS_LCONT
0023& CC 0384&                      JMP       GNLS_DONE
                  GNLS_LCONT
0026& 08 00* 03                     BRCLR     reset_TMT,mag_flags,GNLS_NTMTRST
0029& CD 0000*                      JSR       TMT_RESET      ;Go abort TMT sequence
                  GNLS_NTMTRST
002C& B6 00*                        LDA       uplnk_stat
002E& A4 F0                         AND       #UPLNK_CLR_MSK
0030& B7 00*                        STA       uplnk_stat     ;Mask error bits in uplink status
0032& 05 00* 0D                     BRCLR     TELBFINT,IRQREG,GNLS_NOVF      ;Has downlink overflow occurred?
0035& AE 04                         LDX       #TELBFINT_MSK
0037& BF 00*                        STX       IRQREG                         ;Clear TELBF interrupts
0039& 16 00*                        BSET      up_stat_pnd,uplink_flags       ;Set status uplink pending
003B& AA 09                         ORA       #DNLK_OVF_ERR
003D& B7 00*                        STA       uplnk_stat                     ;Set and store Overflow error
003F& CC 0350&                      JMP       GNLS_UPLNK
                  ;-------------------------------------------------------------------
                  ;@     ELSE
                  ;@       BEGIN
                  ;@         (* No downlink overflow *)
                  ;@         CALLM VALIDATE_MSG;
                  ;@       END;
                  ;-------------------------------------------------------------------
                  GNLS_NOVF
                                    ;VALIDATE_MSG
                  ;-------------------------------------------------------------------
                  ;@     (* Request event time to be latched (write any value)
                  ;@        NOTE: event time takes 0.244msec to be latched *)
                  ;@     EVENTIME := 0;
                  ;@
                  ;@     IF ((up_RAM_pnd of uplink_flags)
                  ;@        OR (intrrg_pnd of uplink_flags)) THEN
                  ;@     BEGIN
                  ;@       (* Only allow RAM uplink if the pacing interval is above
                  ;@          HIGH_RATE, otherwise clear uplink status flag.        *)
                  ;@       IF (timeout_int < HIGH_RATE) THEN
                  ;@         BEGIN
                  ;@           up_RAM_pnd of uplink_flags := FALSE;
                  ;@           intrrg_pnd of uplink_flags := FALSE;
                  ;@           up_stat_pnd of uplink_flags := TRUE;
                  ;@         END;
                  ;@       ELSE
                  ;@         up_stat_pnd of uplink_flags := FALSE;
                  ;@     END;
```

```
                 ;-----------------------------------------------------------------
                 GNLS_UPLNK
0350& B7 00*                 STA     EVENTIME        ;Latch event time count
                 GNLS_UPEVNT
0352& B6 00*                 LDA     uplink_flags
0354& A4 14                  AND     #((1 SHL up_RAM_pnd) + (1 SHL intrrg_pnd))
0356& 27 11                  BEQ     GNLS_NRAMUP     ;Jump if no RAM of interrogate uplink
0358& C6 0000*               LDA     timeout_int
035B& A1 23                  CMP     #HIGH_RATE      ;Is timeout less then upper rate limit?
035D& 24 08                  BHS     GNLS_RTLO       ; No, set uplink status flag false
035F& 15 00*                 BCLR    up_RAM_pnd,uplink_flags
0361& 19 00*                 BCLR    intrrg_pnd,uplink_flags
0363& 16 00*                 BSET    up_stat_pnd,uplink_flags
0365& 20 02                  BRA     GNLS_NRAMUP
                 GNLS_RTLO
0367& 17 00*                 BCLR    up_stat_pnd,uplink_flags
                 GNLS_NRAMUP
                 ;-----------------------------------------------------------------
                 ;@      (* If IPG in VVT mode switch to VVI mode until next event
                 ;@         and scedule uplink if there is enough time.          *)
                 ;@      triggered_mode of PACEMODE := FALSE;
                 ;@      a := timeout_int - EVENTIME
                 ;@      IF (((a > UPSTAT_DELAY) AND (up_stat_pnd of uplink_flags))
                 ;@           OR (a > UPLINK_DELAY)) THEN
                 ;@        CALL SCHEDULE_UPLINK WITHIN R2LIB;
                 ;@      ELSE
                 ;@        uplnk_cnfrm of tlm2_flags := TRUE;
                 ;@
                 ;-----------------------------------------------------------------
0369& 15 00*                 BCLR    triggered_mode,PACEMODE  ;Set in non-VVT mode
036B& C6 0000*               LDA     timeout_int
036E& B0 00*                 SUB     EVENTIME        ;Determine time remaining before next event
0370& A1 1E                  CMP     #UPLINK_DELAY   ;Enough time for block uplink?
0372& 22 07                  BHI     GNLS_SU         ; Yes, then schedule uplink
0374& 07 00* 09              BRCLR   up_stat_pnd,uplink_flags,GNLS_NUPLNK
0377& A1 03                  CMP     #UPSTAT_DELAY   ;Enough time for status uplink?
0379& 23 05                  BLS     GNLS_NUPLNK     ; No, don't attempt uplink
                 GNLS_SU
037B& CD 0000*               JSR     SCHEDULE_UPLINK
037E& 20 02                  BRA     GNLS_CTLBF
                 GNLS_NUPLNK
0380& 18 00*                 BSET    uplnk_cnfrm,tlm2_flags   ;Indicate uplink to follow next event
                 ;-----------------------------------------------------------------
                 ;@      (* Enable TELBF interrupts and clear ADC interrupts *)
                 ;@      TELBFINT of ipgstate_msk := FALSE;
                 ;@    END;
                 ;@
                 ;@  IRQREG := ADCINT_MSK;
                 ;-----------------------------------------------------------------
                 GNLS_CTLBF
0382& 15 00*                 BCLR    TELBFINT,ipgstate_msk
                 GNLS_DONE
0384& A6 01                  LDA     #ADCINT_MSK
0386& B7 00*                 STA     IRQREG          ;Clear pending ADC interrupts
                 GNLS_END
0388& 81                     RTS
                 ;-----------------------------------------------------------------
                 ;@END;    (* GNLSINT_PROC *)
                 ;-----------------------------------------------------------------

                 $EJECT
```

```
                 ;@************************ TELBFINT_PROC ******************************
                 ;@*                                                                  *
                 ;@*  This procedure is the telemetry buffer interrupt handler. It is *
                 ;@*  non-preemptive. It is responsible for scheduling pending uplinks *
                 ;@*  (i.e. markers). If the last uplink was a RAM uplink, all pending *
                 ;@*  uplinks are cancelled. Otherwise, if there is either a pending  *
                 ;@*  interrogate block or measured value, they are uplinked.         *
                 ;@*                                                                  *
                 ;@*  ENTRY CONDITIONS:                                               *
                 ;@*    No other interrupts are allowed during this routine, ADC must *
                 ;@*    be cleared if one occurred during uplink reschedule, and      *
                 ;@*    processing of GAIN/LOSS must wait until after uplink TELBF    *
                 ;@*    completes to insure that the uplink flags are not corrupted   *
                 ;@*                                                                  *
                 ;@*  EXIT CONDITIONS:                                                *
                 ;@*    None.                                                         *
                 ;@*                                                                  *
                 ;@*******************************************************************

                 ;-------------------------------------------------------------------
                 ;@PROCEDURE TELBFINT_PROC;
                 ;@BEGIN
                 ;@
                 ;@  (* If RAM uplink complete clear all pending uplinks *)
                 ;@  IF (uplnk_disabled of uplink_flags) THEN
                 ;@    uplink_flags := 0;
                 ;-------------------------------------------------------------------
                 TELBFINT_PROC
0423& 01 00* 05                  BRCLR    uplnk_disabled,uplink_flags,TLBF_UPLNK
0426& 4F                         CLRA
0427& B7 00*                     STA      uplink_flags    ;Clear all pending uplinks
0429& 20 68                      BRA      TLBF_DONE
                 ;-------------------------------------------------------------------
                 ;@  ELSE
                 ;@    BEGIN
                 ;@      (* Previous uplink was not a RAM uplink, uplink pending *)
                 ;@      IF (mrkr_pnd of uplink_flags) THEN
                 ;@        BEGIN
                 ;@          (* Marker from POS is pending *)
                 ;@          mrkr_pnd of uplink_flags := FALSE;
                 ;@          TELADHI := HIADDR(marker_val[0]);
                 ;@          TELADLO := LOADDR(marker_val[0]);
                 ;@          BYTCOUNT := marker_cnt;
                 ;@          ULID := MARKER_ID;
                 ;@          RAM_uplink of TELSTAT := TRUE;
                 ;@        END;
                 ;@
                 ;-------------------------------------------------------------------
                 TLBF_UPLNK
042B& 0D 00* 11                  BRCLR    mrkr_pnd,uplink_flags,TLBF_LCAP
042E& 1D 00*                     BCLR     mrkr_pnd,uplink_flags
0430& A6 ..X                     LDA      #HIGH marker_val    ;Load register with hi address of marker value
                                                                address
0432& B7 00*                     STA      TELADHI
0434& A6 ..X                     LDA      #LOW marker_val     ;Load register with low address of marker value
                                                                address
0436& B7 00*                     STA      TELADLO
0438& CE 0000*                   LDX      marker_cnt          ;Load x with byte count
043B& A6 43                      LDA      #MARKER_ID          ;Load a with marker identification byte
043D& 20 49                      BRA      TLBF_STRTU
                 ;-------------------------------------------------------------------
                 ;@      ELSE IF (lcap_mrkr_pnd of uplink_flags) THEN
                 ;@        BEGIN
                 ;@          (* Marker from loss of capture is pending *)
                 ;@          lcap_mrkr_pnd of uplink_flags := FALSE;
                 ;@          TELADHI := HIADDR(lcap_marker);
                 ;@          TELADLO := LOADDR(lcap_marker);
                 ;@          BYTCOUNT := 1;
```

```
                   ;@          ULID := MARKER_ID;
                   ;@          RAM_uplink of TELSTAT := TRUE;
                   ;@        END;
                   ;@          :
                   ;-----------------------------------------------------------------
                   TLBF_LCAP
043F& 08 00* 10                BRCLR    lcap_mrkr_pnd,uplink_flags,TLBF_INTRRG
0442& 1B 00*                   BCLR     lcap_mrkr_pnd,uplink_flags
0444& A6 ..X                   LDA      #HIGH lcap_marker   ;Load register with hi address of lcap marker v
                                                                 alue address
0446& B7 00*                   STA      TELADHI
0448& A6 ..X                   LDA      #LOW lcap_marker    ;Load register with low address byte of lcap ma
                                                                 rker value address
044A& B7 00*                   STA      TELADLO
044C& AE 01                    LDX      #1                  :Load x with byte count
044E& A6 43                    LDA      #MARKER_ID          ;Load a with marker identification byte
0450& 20 36                    BRA      TLBF_STRTU
                   ;-----------------------------------------------------------------
                   ;@       ELSE IF (intrrg_pnd of uplink_flags) THEN
                   ;@         BEGIN
                   ;@           intrrg_pnd of uplink_flags := FALSE;
                   ;@           statbyt := uplnk_stat;
                   ;@           CALLM LOAD_INTRRG_UPLINK WITHIN R2LIB;
                   ;@           RAM_uplink of TELSTAT := TRUE;
                   ;@         END;
                   ;-----------------------------------------------------------------
                   TLBF_INTRRG
0452& 09 00* 20                BRCLR    intrrg_pnd,uplink_flags,TLBF_MEAS
0455& 19 00*                   BCLR     intrrg_pnd,uplink_flags
0457& B6 00*                   LDA      uplnk_stat
0459& C7 0000*                 STA      statbyt         ;Update status byte
                   TLBF_LDIN
                               ;LOAD_INTRRG_UPLINK
                   TLBF_LDIN_END
0471& 16 00*                   BSET     RAM_uplink,TELSTAT  ;Initiate uplink
0473& 20 1E                    BRA      TLBF_DONE
                   ;-----------------------------------------------------------------
                   ;@       ELSE IF (meas_pnd of uplink_flags) THEN
                   ;@         BEGIN
                   ;@           meas_pnd of uplink_flags := FALSE;
                   ;@           TELADHI := HIADDR(meas_val[0]);
                   ;@           TELADLO := LOADDR(meas_val[0]);
                   ;@           BYTCOUNT := meas_count;
                   ;@           ULID := meas_id;
                   ;@           RAM_uplink of TELSTAT := TRUE;
                   ;@         END;
                   ;-----------------------------------------------------------------
                   TLBF_MEAS
0475& 0F 00* 18                BRCLR    meas_pnd,uplink_flags,TLBF_NUPLNK
0478& 1F 00*                   BCLR     meas_pnd,uplink_flags
047A& A6 ..X                   LDA      #HIGH meas_val      ;Load register with hi address of measured valu
                                                                 e address
047C& B7 00*                   STA      TELADHI
047E& A6 ..X                   LDA      #LOW meas_val       ;Load register with low address byte of measure
                                                                 d value address
0480& B7 00*                   STA      TELADLO
0482& CE 0000*                 LDX      meas_count          ;Load x with byte count
0485& C6 0000*                 LDA      meas_id             ;Load a with marker identification byte
                   TLBF_STRTU
0488& BF 00*                   STX      BYTCOUNT            ;Store byte count
048A& B7 00*                   STA      ULID                ;Store marker identification byte
048C& 16 00*                   BSET     RAM_uplink,TELSTAT  ;Set the telemetry status byte and exit
048E& 20 03                    BRA      TLBF_DONE
                   ;-----------------------------------------------------------------
                   ;@       ELSE (* No pending uplinks *)
                   ;@         uplink_flags := 0;
                   ;@     END;
                   ;@  (* Clear pending ADC interrupts *)
```

```
Avocet 6805 Assembler v2.20, #01002   Chip=146805
========== R2 TELEMETRY MODULE ================== File: TLM.ASM              1/18/90 10:35:56
========== $Revision:   3.3  $ =============                                        Page 40

                  ;@  IRQREG := ADCINT_MSK;
                  ;@
                  ;-----------------------------------------------------------------
                  TLBF_NUPLNK
0490& 4F                          CLRA
0491& B7 00*                      STA        uplink_flags    ;Clear uplink flags, no uplinks pending
                  TLBF_DONE
0493& A6 01                       LDA        #ADCINT_MSK
0495& B7 00*                      STA        IRQREG          ;Clear pending ADC interrupts
                  TLBF_END
0497& 81                          RTS
                  ;-----------------------------------------------------------------
                  ;@END;     (* TELBFINT_PROC *)
                  ;@
                  ;-----------------------------------------------------------------
                                  END
```

```
;@********************** LOAD_INTRRG_UPLINK ****************************
;@*                                                                     *
;@*  This macro loads the telemetry registers in preparation for an    *
;@*  Interrogate block uplink.                                         *
;@*                                                                     *
;@*  ENTRY CONDITIONS:                                                 *
;@*    Uplink data registers are ready to be loaded without conflict.  *
;@*                                                                     *
;@*  EXIT CONDITIONS:                                                  *
;@*    The interrogate block the size of INTRRG_SIZ and starting at    *
;@*    the address pointed to by INTRRG_AD is setup for uplink.        *
;@*                                                                     *
;@**********************************************************************

;-----------------------------------------------------------------------
;@  MACRO LOAD_INTRRG_UPLINK;
;@  BEGIN
;@
;@     (* Load interrogate status byte *)
;@     intrrg_R2_stat := R2_stat;
;@     (* Uplink channel assumed free and upink_disabled bit set *)
;@     TELADHI := HIBYTE(INTRRG_AD);
;@     TELADLO := LOWBYTE(INTRRG_AD);
;@     BYTCOUNT := INTRRG_SIZ;
;@     ULID := RAM_ID;
;@
;@  END;    (* LOAD_INTRRG_UPLINK *)
;-----------------------------------------------------------------------
;LOAD_INTRRG_UPLINK %MACRO
;               LDA     r2_stat         ;Get r2 status byte
;               STA     intrrg_r2_stat  ;put in interrogate status byte
;               LDA     #HIGH INTRRG_AD ;Get address hi byte
;               STA     TELADHI         ;Send it to the hardware
;
;               LDA     #LOW INTRRG_AD  ;Get address lo byte
;               STA     TELADLO         ;Send it to the hardware
;
;               LDA     #INTRRG_SIZ     ;Get byte count
;               STA     BYTCOUNT        ;Write hardware register
;               LDA     #RAM_ID         ;Get ID
;               STA     ULID            ; etc. etc. etc.
;               %ENDM

$EJECT
```

```
Avocet 6805 Assembler v2.20, #01002   Chip=146805                                 1/08/90 11:38:20
========== R2 LIBRARY MODULE ================== File: R2LIB.ASM                          Page 8
========== $Revision:   3.3  $ =============

;@*********************** LOAD_RAM_UPLINK ****************************
;@*                                                                   *
;@*  This macro loads the telemetry registers in preparation for a    *
;@*  RAM block uplink.                                                *
;@*                                                                   *
;@*  ENTRY CONDITIONS:                                                *
;@*    Uplink data registers are ready to be loaded without conflict. *
;@*                                                                   *
;@*  EXIT CONDITIONS:                                                 *
;@*    A RAM block of length indicated by P_rd_bytes starting at the  *
;@*    address indicated by P_rd_start is setup for uplink.           *
;@*                                                                   *
;@*********************************************************************

;----------------------------------------------------------------------
;@  MACRO LOAD_RAM_UPLINK;
;@  BEGIN
;@
;@    (* Uplink channel assumed free and uplnk_disabled bit set *)
;@    intrrg_R2_stat := R2_stat;
;@    TELADHI := HIBYTE(P_rd_start);
;@    TELADLO := LOWBYTE(P_rd_start);
;@    BYTCOUNT := P_rd_bytes;
;@    ULID := RAM_ID;
;@
;@  END;    (* LOAD_RAM_UPLINK *)
;----------------------------------------------------------------------
;LOAD_RAM_UPLINK %MACRO
;               LDA     r2_stat         ;Get r2 status byte
;               STA     intrrg_r2_stat  ;put in interrogate status byte.
;               LDA     P_rd_start      ;Get address hi byte
;               STA     TELADHI         ;Send it to the hardware
;
;               LDA     P_rd_start +1   ;Get address lo byte
;               STA     TELADLO         ;Send it to the hardware
;
;               LDA     P_rd_bytes      ;Get byte count
;               STA     BYTCOUNT        ;Write hardware register
;               LDA     #RAM_ID         ;Get ID
;               STA     ULID            ; etc. etc. etc.
;               %ENDM


$RESETLN

$NOALLPUBLIC
$NOLIST                                 ;Don't List the equate file
```

```
                     ;@*********************** SCHEDULE_UPLINK ****************************
                     ;@*                                                                  *
                     ;@*  This procedure schedules uplink of RAM, interrogate block, or   *
                     ;@*  status in this order of priority.                               *
                     ;@*                                                                  *
                     ;@*  ENTRY CONDITIONS:                                               *
                     ;@*    No other interrupts are allowed during this routine, ADC      *
                     ;@*    interrupts must be cleared if one occurred during uplink      *
                     ;@*    scheduling. Processing of the GAIN/LOSS and TLBF interrupts   *
                     ;@*    wait until after uplink is scheduled to ensure that the       *
                     ;@*    uplink flags are not corrupted.                               *
                     ;@*                                                                  *
                     ;@*  EXIT CONDITIONS:                                                *
                     ;@*    Either a RAM block, an Interrogate block, or a status         *
                     ;@*    confirmation block are uplinked if any are pending.           *
                     ;@*    Status is imbedded in a RAM or Interrogate block uplink.      *
                     ;@*                                                                  *
                     ;@********************************************************************

                     ;--------------------------------------------------------------------
                     ;@  PROCEDURE SCHEDULE_UPLINK;
                     ;@  BEGIN
                     ;@
                     ;@    (* Load status byte for RAM uplink and the load telemetry
                     ;@       registers for uplink. *)
                     ;@    IF (up_RAM_pnd of uplink_flags) THEN
                     ;@      BEGIN
                     ;@        (* Load for Ram uplink *)
                     ;@        CALLM LOAD_RAM_UPLINK;
                     ;@        up_RAM_pnd of uplink_flags := FALSE;
                     ;@      END;
                     ;--------------------------------------------------------------------
                     SCHEDULE_UPLINK
                                                             ;Jump if NOT RAM uplink
0180& 05 00* 1D                  BRCLR   up_RAM_pnd,uplink_flags,SUP_INTRRG
                     SU_LRU
                                 ;LOAD_RAM_UPLINK
0183& B6 00*                     LDA     r2_stat          ;Get r2 status byte
0185& C7 0000*                   STA     intrrg_r2_stat   ;put in interrogate status byte
0188& C6 0000*                   LDA     P_rd_start       ;Get address hi byte
018B& B7 00*                     STA     TELADHI          ;Send it to the hardware

018D& C6 ....X                   LDA     P_rd_start +1    ;Get address lo byte
0190& B7 00*                     STA     TELADLO          ;Send it to the hardware

0192& C6 0000*                   LDA     P_rd_bytes       ;Get byte count
0195& B7 00*                     STA     BYTCOUNT         ;Write hardware register
0197& A6 C0                      LDA     #RAM_ID          ;Get ID
0199& B7 00*                     STA     ULID             ; etc. etc. etc.
                     SU_LRU_END
019B& 15 00*                     BCLR    up_RAM_pnd,uplink_flags ;Clear the pending flag
019D& CC 01D1&                   JMP     SUP_STRT         ;Go start uplink

                     ;--------------------------------------------------------------------
                     ;@    ELSE IF (intrrg_pnd of uplink_flags) THEN
                     ;@      BEGIN
                     ;@        (* Load for interrogate block uplink *)
                     ;@        CALLM LOAD_INTRRG_UPLINK WITHIN R2LIB;
                     ;@        intrrg_pnd of uplink_flags := FALSE;
                     ;@      END;
                     ;--------------------------------------------------------------------
                     SUP_INTRRG
                                                             ;Jump if NOT interrogate
01A0& 09 00* 1A                  BRCLR   intrrg_pnd,uplink_flags,SUP_STAT
                     SU_LIU
                                 ;LOAD_INTRRG_UPLINK
01A3& B6 00*                     LDA     r2_stat          ;Get r2 status byte
01A5& C7 0000*                   STA     intrrg_r2_stat   ;put in interrogate status byte
```

```
01A8& A6 ..X                     LDA     #HIGH INTRRG_AD ;Get address hi byte
01AA& B7 00*                     STA     TELADHI         ;Send it to the hardware

01AC& A6 ..X                     LDA     #LOW INTRRG_AD  ;Get address lo byte
01AE& B7 00*                     STA     TELADLO         ;Send it to the hardware

01B0& A6 00*                     LDA     #INTRRG_SIZ     ;Get byte count
01B2& B7 00*                     STA     BYTCOUNT        ;Write hardware register
01B4& A6 C0                      LDA     #RAM_ID         ;Get ID
01B6& B7 00*                     STA     ULID            ; etc. etc. etc.
                     SU_LIU_END
01B8& 19 00*                     BCLR    intrrg_pnd,uplink_flags ;Clear the flag
01BA& CC 01D1&                   JMP     SUP_STRT        ;Go start uplink

                     ;----------------------------------------------------------------
                     ;@     ELSE IF (up_stat_pnd of uplink_flags) THEN
                     ;@       BEGIN
                     ;@         (* Load for status ID byte for uplink *)
                     ;@         ULID := STATUS_ID;
                     ;@         up_stat_pnd of uplink_flags := FALSE;
                     ;@       END;
                     ;----------------------------------------------------------------
                     SUP_STAT
                                                         ;Jump if NOT status ID byte
01BD& 07 00* 08                  BRCLR   up_stat_pnd,uplink_flags,SUP_NO_UP
01C0& A6 80                      LDA     #STATUS_ID
01C2& B7 00*                     STA     ULID            ;Write status ID to hardware
01C4& 17 00*                     BCLR    up_stat_pnd,uplink_flags ;Clear the flag
01C6& 20 09                      BRA     SUP_STRT        ;Go start Uplink

                     ;----------------------------------------------------------------
                     ;@     ELSE
                     ;@       BEGIN
                     ;@         (* No uplink scheduled reset telemetry and exit routine *)
                     ;@         uplink_flags := 0;
                     ;@         CALL SET_TLM_TYPE
                     ;@         downlink_enable of TELSTAT := TRUE;
                     ;@         EXIT;
                     ;@       END;
                     ;----------------------------------------------------------------
                     SUP_NO_UP
01C8& 3F 00*                     CLR     uplink_flags    ;Clear uplink_flags, no uplink
01CA& CD 01DE&                   JSR     SET_TLM_TYPE    ;Set telemetry type and enable downlink
01CD& 1A 00*                     BSET    downlink_enabled,TELSTAT
01CF& 20 0C                      BRA     SUP_END         ;Go exit

                     ;----------------------------------------------------------------
                     ;@     (* Set telemetry type start uplink and enable downlink *)
                     ;@     statbyt := uplnk_stat;
                     ;@     CALL SET_TLM_TYPE;
                     ;@     downlink_enable of TELSTAT := TRUE;
                     ;@     RAM_uplink of TELSTAT := TRUE;
                     ;@
                     ;----------------------------------------------------------------
                     SUP_STRT
01D1& B6 00*                     LDA     uplnk_stat      ;Get uplink status
01D3& C7 0000*                   STA     statbyt
01D6& CD 01DE&                   JSR     SET_TLM_TYPE    ;Set telemetry type
01D9& AA 28                      ORA     #(1 SHL RAM_uplink) + (1 SHL downlink_enabled)
01DB& B7 00*                     STA     TELSTAT         ;enable downlink and start uplink
                     SUP_END
01DD& 81                         RTS                     ;Return to caller
                     ;----------------------------------------------------------------
                     ;@   END; (* SCHEDULE_UPLINK *)
                     ;----------------------------------------------------------------

                     $EJECT
```

```
Avocet 6805 Assembler v2.20, #01002   Chip=146805                              1/08/90 11:38:20
========== R2 LIBRARY MODULE ================== File: R2LIB.ASM                       Page 31
========== $Revision:   3.3  $ =============

                ;@*********************** SET_TLM_TYPE ******************************
                ;@*                                                                   *
                ;@*  This procedure decodes the telemetry type in P_tlm_type and      *
                ;@*  sets up the hardware and marker channel accordingly.             *
                ;@*                                                                   *
                ;@*  ENTRY COND. :                                                    *
                ;@*    P_tlm_type contains the desired telemetry.                     *
                ;@*                                                                   *
                ;@*  EXIT COND. :                                                     *
                ;@*    The analog uplink telemetry is updated on the next frame.      *
                ;@*    Curr_tlm_type is written to PACESTAT and may not equal         *
                ;@*    P_tlm_type.                                                    *
                ;@*    a - contains the current value of the TELSTAT register.        *
                ;@*
                ;@*******************************************************************

                ;-------------------------------------------------------------------
                ;@   PROCEDURE SET_TLM_TYPE;
                ;@   BEGIN
                ;@
                ;@     (* test for markers uplink selected *)
                ;@     IF (marker_enabled of P_tlm_type := TRUE) THEN
                ;@       marker_active of mag_flags := TRUE;
                ;@     ELSE
                ;@       marker_active of mag_flags := FALSE;
                ;@
                ;-------------------------------------------------------------------
                SET_TLM_TYPE
01DE& C6 0000*          LDA     P_tlm_type      ;Jump if idle markers set
01E1& A4 01             AND     #(1 SHL marker_enabled)
01E3& 27 04             BEQ     STT_ICLR
01E5& 1E 00*            BSET    markers_active,mag_flags ;Show idle markers
01E7& 20 02             BRA     STT_ADJ         ;Go adjust telem type
                STT_ICLR
01E9& 1F 00*            BCLR    markers_active,mag_flags

                ;-------------------------------------------------------------------
                ;@     (* adjust the telemetry type *)
                ;@     curr_tlm_type := (P_tlm_type AND TLM_TYPE_MSK) OR IDLE_UPLINK;
                ;@     TELSTAT := (TELSTAT AND TELSTAT_MSK) OR curr_tlm_type;
                ;@
                ;-------------------------------------------------------------------
                STT_ADJ
01EB& C6 0000*          LDA     P_tlm_type      ;Get telemetry type
01EE& A4 C6             AND     #TLM_TYPE_MSK   ;Isolate real time uplink type
01F0& AA 01             ORA     #IDLE_UPLINK    ;Set uplink idle bit and save as current type
01F2& B7 00*            STA     curr_tlm_type
01F4& B6 00*            LDA     TELSTAT         ;Get current value of TELSTAT
01F6& A4 38             AND     #TELSTAT_MSK    ; and mask changeable bits
01F8& BA 00*            ORA     curr_tlm_type   ;Set new uplink type
01FA& B7 00*            STA     TELSTAT         ;Write new TELSTAT and return
                STT_END
01FC& 81                RTS
                ;-------------------------------------------------------------------
                ;@   END;    (* SET_TLM_TYPE *)
                ;-------------------------------------------------------------------

                $EJECT
```

WHAT IS CLAIMED IS:

1. A method for transmitting data percutaneously between a medical device implanted within a human body and an external device, comprising the steps of:

(a) formatting the data to be transmitted by:

(1) establishing a frame having a fixed time interval;

(2) placing a unique synchronizing signal at a first fixed range within said frame;

(3) placing a frame identifier at a second fixed range within said frame; and

(4) placing said data at a third fixed range within said frame; and

(b) transmitting said formatted data between said implanted medical device and said external device.

2. A method according to claim 1, wherein said data is representative of more than one type of data, and wherein said frame identifier is indicative of the data type within said frame being transmitted.

3. A method according to claim 2, wherein said data is in digital format.

4. A method according to claim 3, wherein each of said steps (a)(2), (a)(3) and (a)(4) thereof further comprises generating a burst of radio frequency energy at a time within the corresponding fixed range appropriate to pulse position modulate said burst.

5. An apparatus for transmitting data percutaneously between an implantable medical device and an external device, comprising:

(a) frame defining means for defining a transmission frame of a fixed time interval;

(b) first means coupled to said frame defining means for transmitting a synchronizing signal within a first time range of said transmission frame;

(c) second means coupled to said frame defining means for transmitting a frame identifier within a second time range of said transmission frame; and

(d) third means coupled to said frame defining means for transmitting said data within a third time range of said transmission frame.

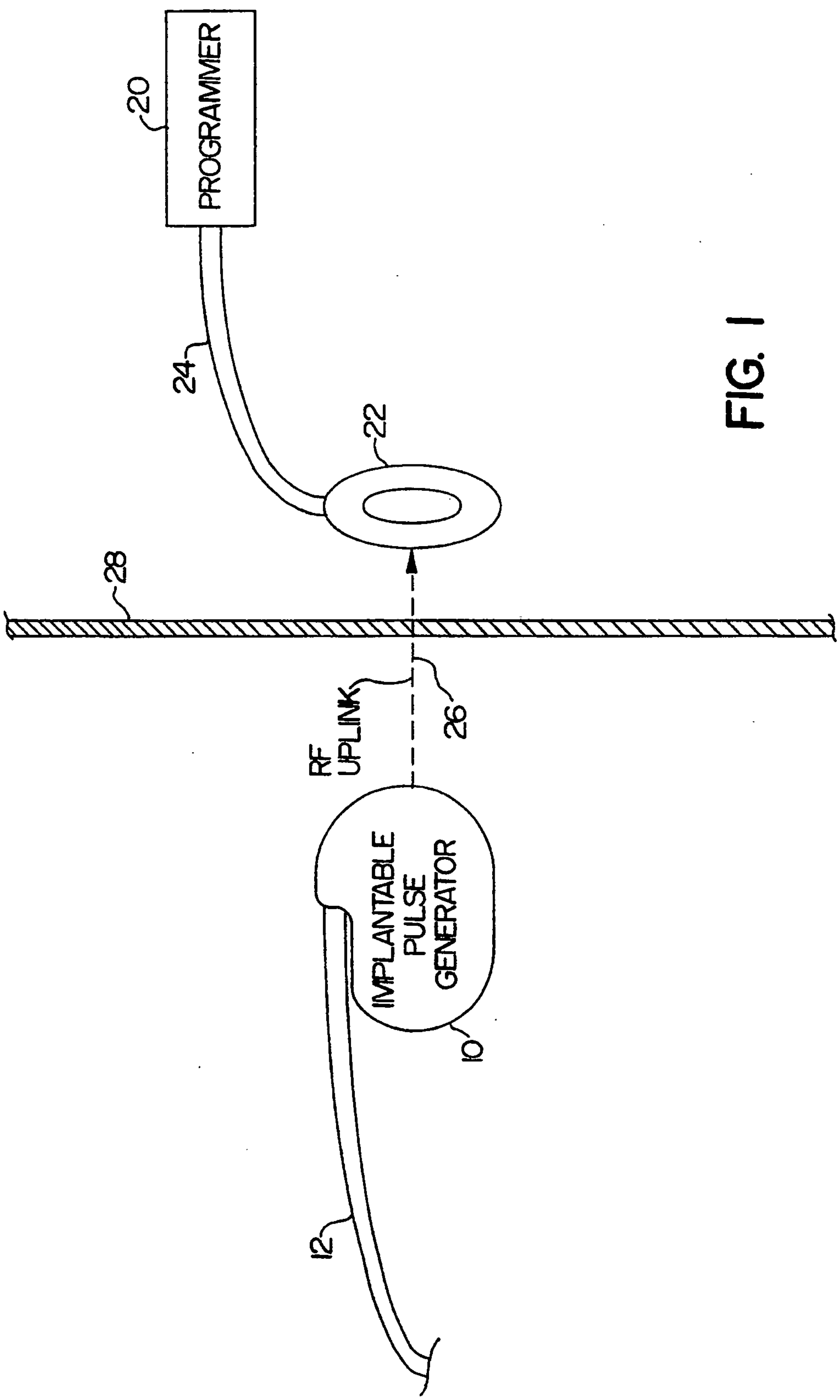

PROGRAMMER
20
24
22
28
RF
UPLINK
26
IMPLANTABLE
PULSE
GENERATOR
10
12

FIG. 1

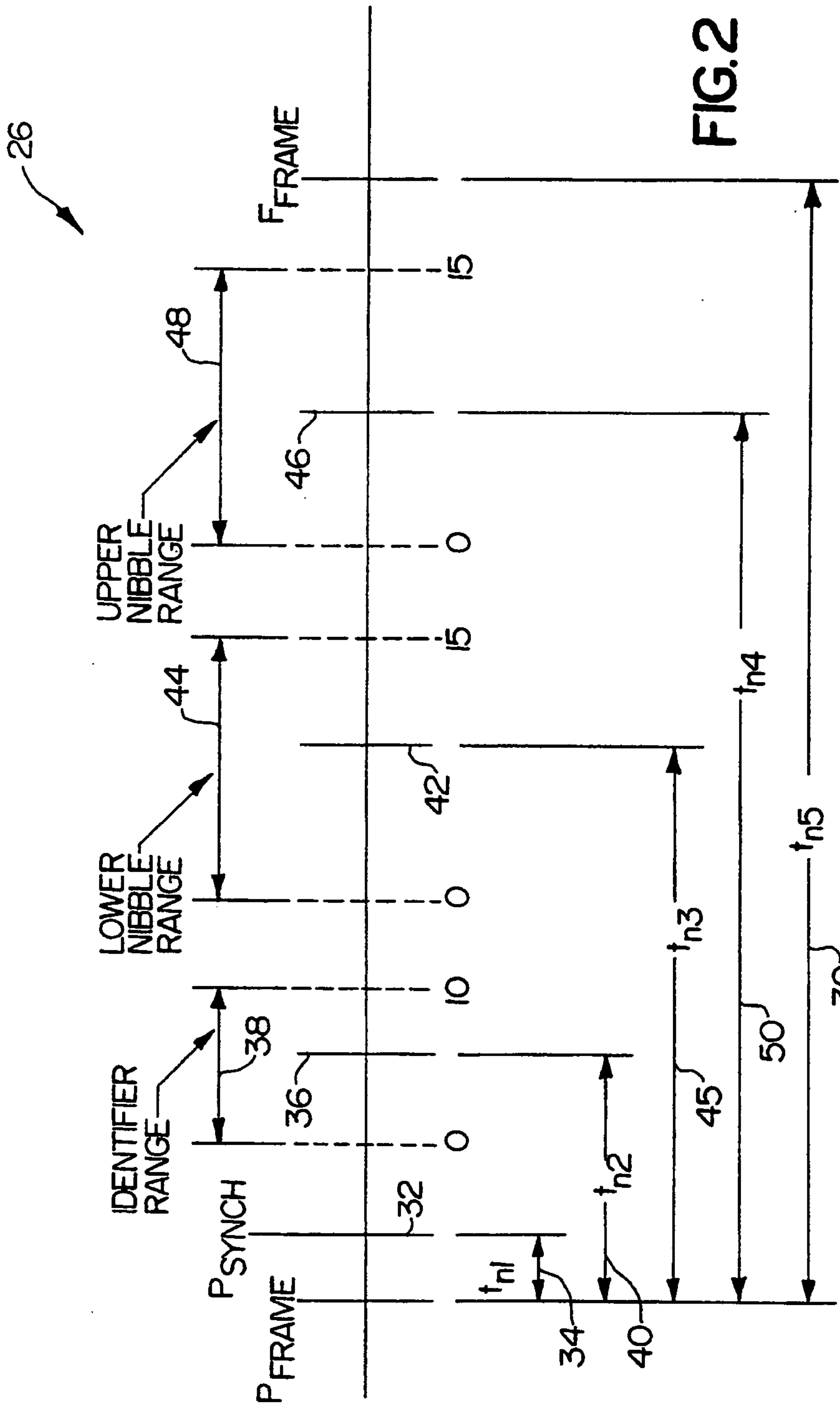
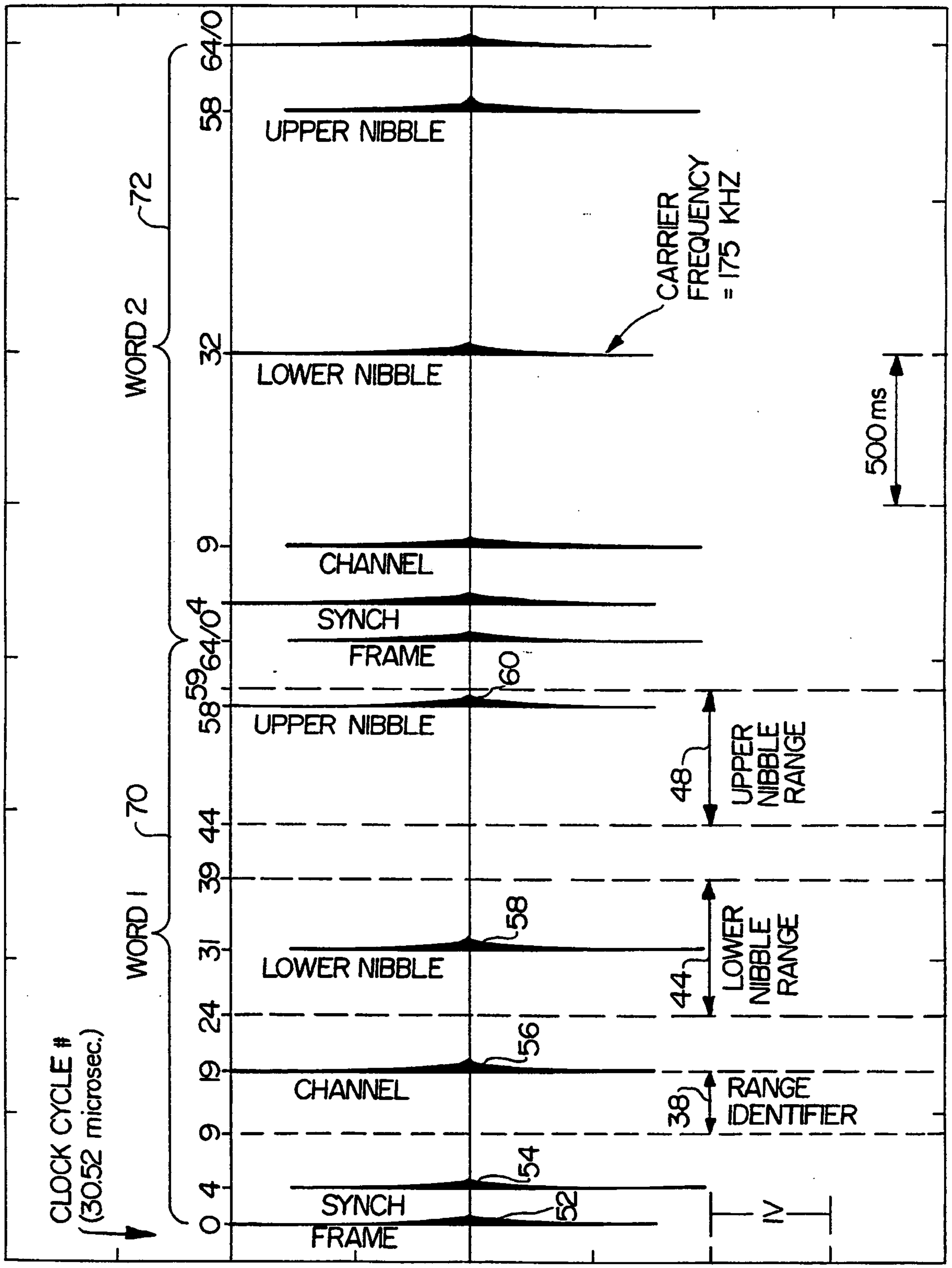

26
F_{FRAME}
15
48
UPPER NIBBLE RANGE
46
0
15
44
LOWER NIBBLE RANGE
42
0
10
38
IDENTIFIER RANGE
36
0
32
P_{SYNCH}
P_{FRAME}
t_{nl}
34
t_{n2}
40
t_{n3}
45
t_{n4}
50
t_{n5}
30

FIG.2

FIG. 3


CLOCK CYCLE # (30.52 microsec.)
WORD 1
70
WORD 2
72
0
4
9
19
24
31
39
44
58
59
64/0
4
9
32
58
64/0
FRAME
SYNCH
CHANNEL
LOWER NIBBLE
UPPER NIBBLE
52
54
56
58
60
38
RANGE IDENTIFIER
44
LOWER NIBBLE RANGE
48
UPPER NIBBLE RANGE
CARRIER FREQUENCY = 175 KHZ
500 ms
≥

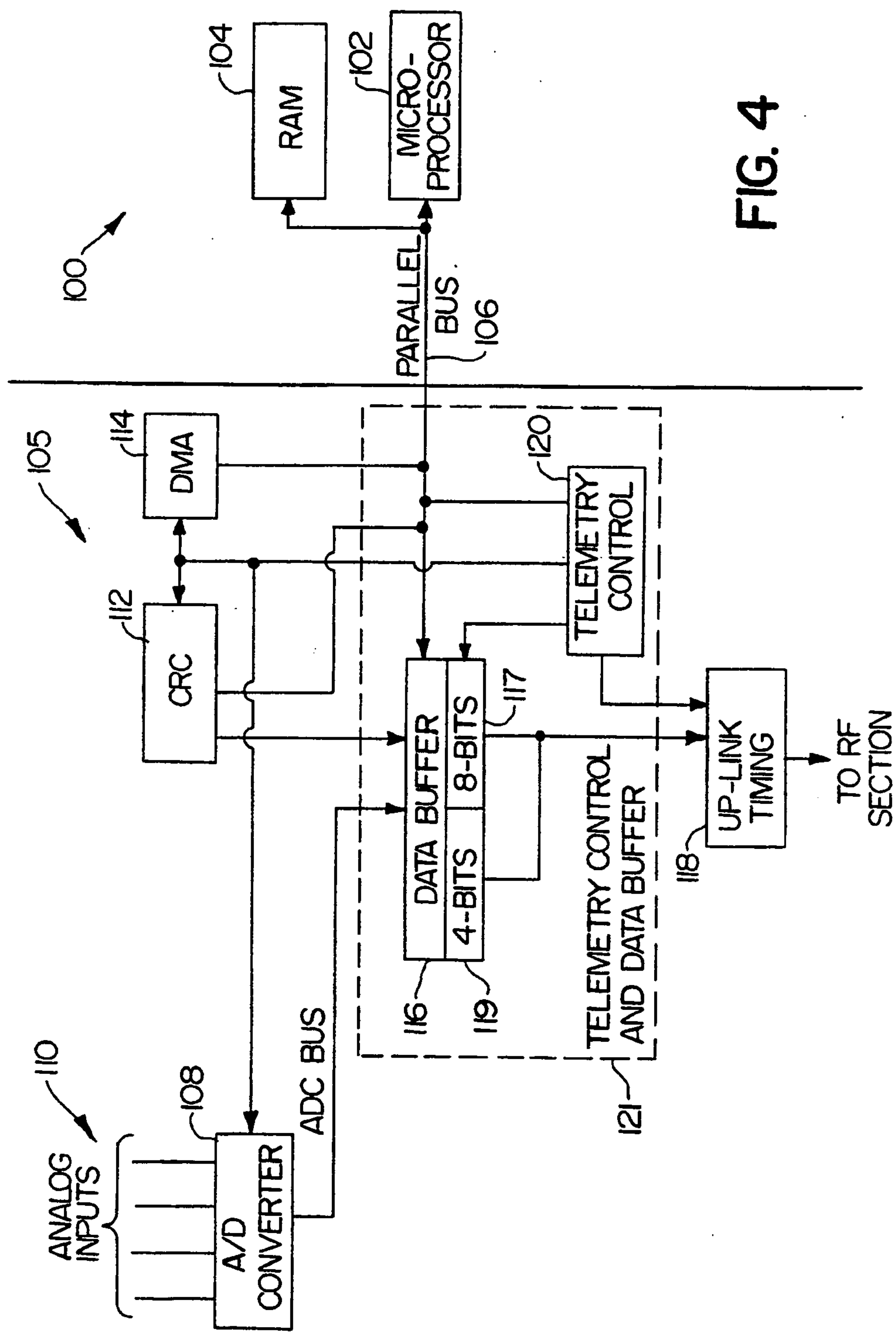

100
RAM
104
MICRO-PROCESSOR
102
PARALLEL BUS
106
105
DMA
114
CRC
112
TELEMETRY CONTROL
120
DATA BUFFER
8-BITS
4-BITS
117
116
119
UP-LINK TIMING
118
TO RF SECTION
TELEMETRY CONTROL AND DATA BUFFER
121
ANALOG INPUTS
110
A/D CONVERTER
108
ADC BUS

FIG. 4

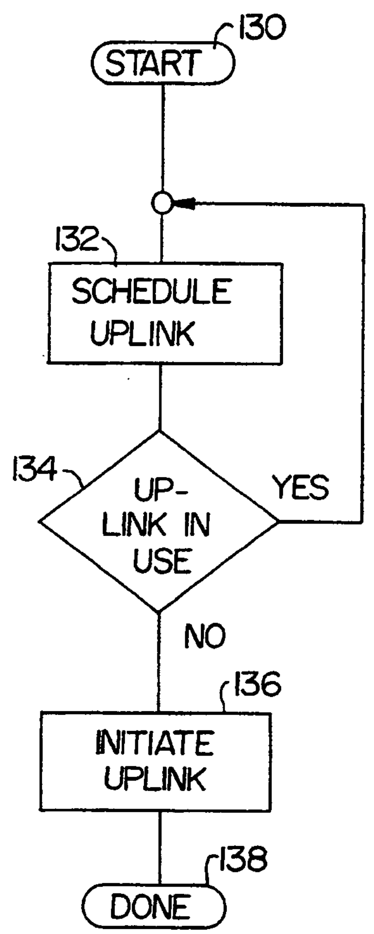

130
START
132
SCHEDULE UPLINK
134
UP-LINK IN USE
YES
NO
136
INITIATE UPLINK
138
DONE

FIG. 5

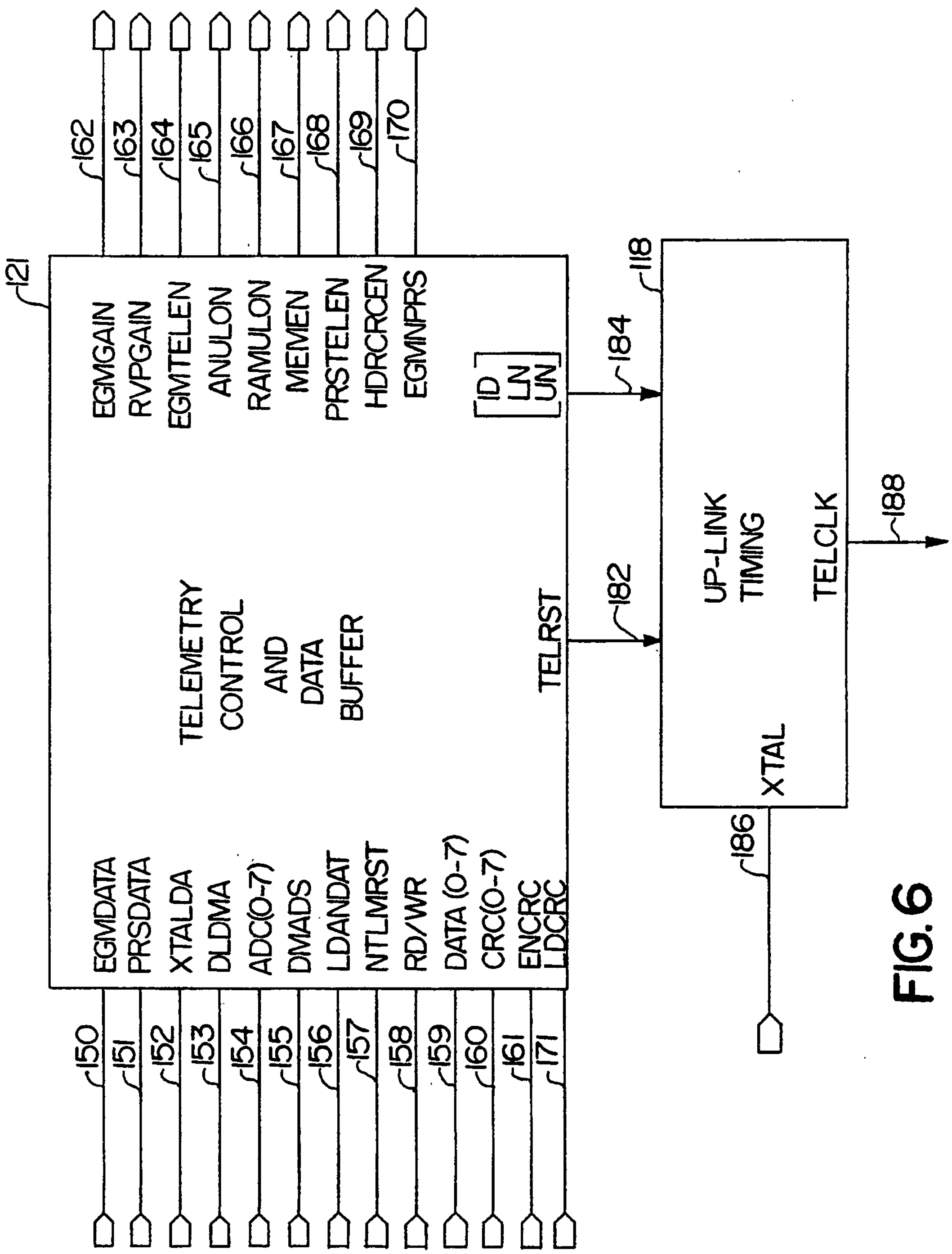

TELEMETRY CONTROL AND DATA BUFFER
121
EGMGAIN
RVPGAIN
EGMTELEN
ANULON
RAMULON
MEMEN
PRSTELEN
HDRCRCEN
EGMNPRS
ID
LN
UN
TELRST
162
163
164
165
166
167
168
169
170
EGMDATA
PRSDATA
XTALDA
DLDMA
ADC(O-7)
DMADS
LDANDAT
NTLMRST
RD/WR
DATA (O-7)
CRC(O-7)
ENCRC
LDCRC
150
151
152
153
154
155
156
157
158
159
160
161
171
184
182
UP-LINK TIMING
118
TELCLK
188
XTAL
186

FIG. 6

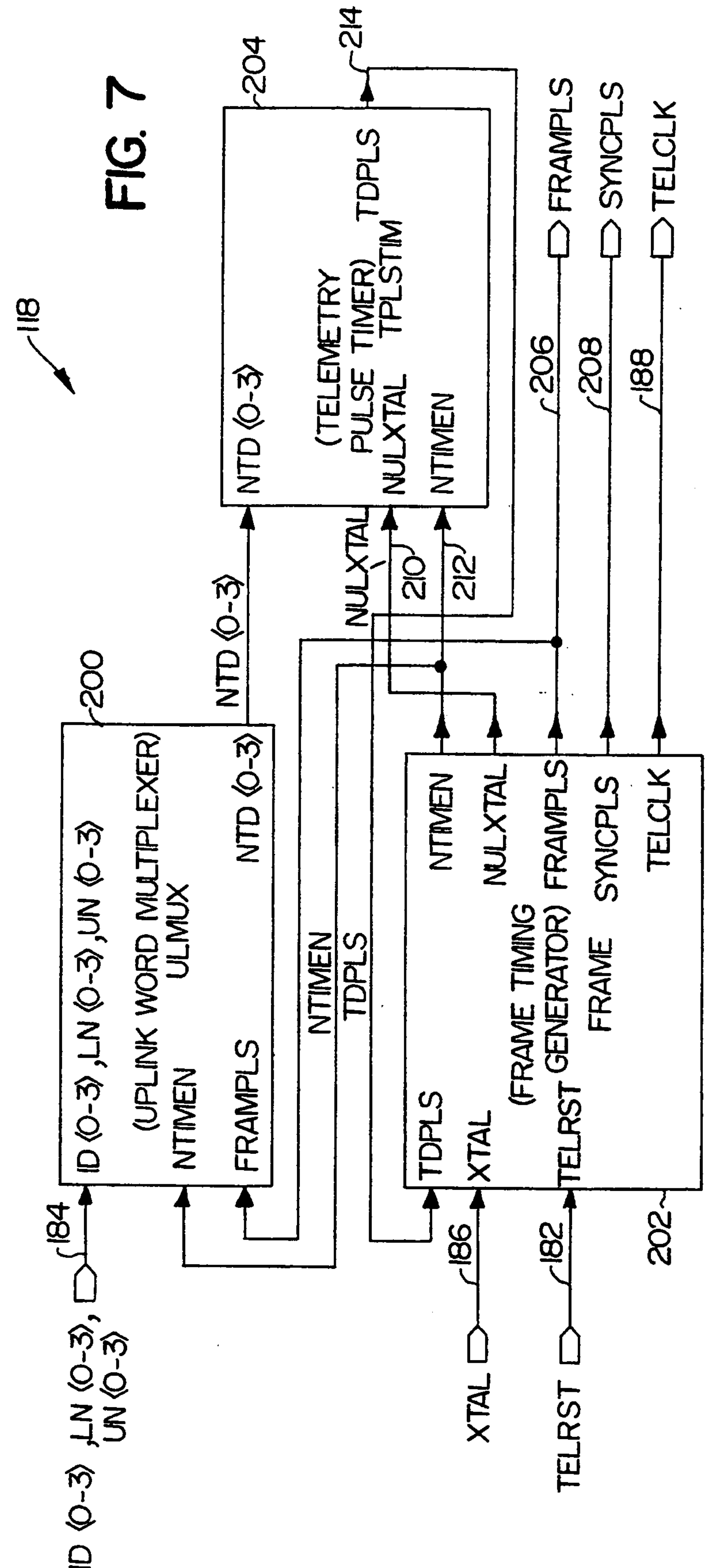

118
204
214
NTD (0-3)
(TELEMETRY PULSE TIMER)
TPLSTIM
NULXTAL
NTIMEN
TDPLS
206
208
188
FRAMPLS
SYNCPLS
TELCLK
NULXTAL
210
212
200
NTD (0-3)
ID (0-3), LN (0-3), UN (0-3)
(UPLINK WORD MULTIPLEXER)
ULMUX
NTIMEN
FRAMPLS
NTD (0-3)
NTIMEN
TDPLS
TDPLS
XTAL
TELRST
(FRAME TIMING GENERATOR)
FRAME
NTIMEN
NULXTAL
FRAMPLS
SYNCPLS
TELCLK
202
184
186
182
ID (0-3), LN (0-3), UN (0-3)
XTAL
TELRST

FIG. 7

ANNEX TO THE INTERNATIONAL SEARCH REPORT
ON INTERNATIONAL PATENT APPLICATION NO.PCT/US 91/00309

SA 44478

This annex lists the patent family members relating to the patent documents cited in the above-mentioned international search report.
The members are as contained in the European Patent Office EDP file on 23/03/91
The European Patent office is in no way liable for theseparticulars which are merely given for the purpose of information.

Patent document cited in search report	Publication date	Patent family member(s)		Publication date
EP-A2- 0071131	09/02/83	AU-D-	8631682	03/02/83
		DE-A-	3130104	17/02/83
		JP-A-	58041570	10/03/83
		US-A-	4524774	25/06/85
DE-C2- 2703700	04/08/83	NONE		
DE-A1- 3119119	09/12/82	NONE		
US-A- 4556063	03/12/85	CA-A-	1183576	05/03/85
		CA-C-	1187140	14/05/85
		DE-A-	3139452	24/06/82
		FR-A-B-	2491659	09/04/82
		JP-A-	57089872	04/06/82
		NL-A-	8104534	03/05/82

For more details about this annex : see fficial Journal of the European patent ffice, No. 12/82